Software Delivery Models

Comparing Approaches and Debunking the Myths

Diversity Limited

Contents

Introduction 2

Comparison Chart: Software Delivery Models 3

An Overview of the Three Major Software Delivery Models 4

- On-Premises Software Deployment 4
- Hosted Software Deployment 5
- Software as a Service (SaaS) 6

The Pros and Cons at a Glance 7

IT and Operational Requirements 8

- Software Procurement 8
- Implementation 9
- IT Operations and Management 10
- Security 11
- Upgrades 12
- Business Continuity 13
- Ease of Customization 14

Service and Support 15

Total Cost of Ownership 16

Conclusion: Software Orientation, Service Orientation 18

About Intacct 19

About Diversity Analysis 20

About the Authors 21

© Copyright 2010 - Diversity Analysis

Introduction

Relentless innovation in the way businesses apply computing power to automate processes has translated into tremendous gains in just a few short decades. The unprecedented benefits we enjoy today are virtually taken for granted - greater velocity, productivity and accuracy – along with lower overall costs and greater operational visibility and insight. A chief contributor to these improvements is the delivery model that businesses adopt for consuming and using computing resources.

The computing industry has evolved from on-premises software implementations to off-site hosting (and its offshoot, the "application service provider" – or ASP – model) to Software as a Service (SaaS). Despite significant differences among these models, a wide range of misperceptions and myths persist about each model's relative merits and potential drawbacks. Some even believe that this market confusion caused by mislabeling and proliferating jargon is a deliberate industry strategy to jockey for a marketing advantage.

The following white paper explains, contrasts and compares the three major software delivery models and provides buyers with useful criteria to evaluate software offerings and determine the suitability of the approaches to their unique environment and needs.

© Copyright 2010 - Diversity Analysis

Comparison Chart: Software Delivery Models

	On-premises software	Hosted software	Software as a Service
Application Development	Developed as "one-size-fits-all" model	Borrows on-premises software and modifies it for online delivery	Developed from the ground up for online delivery
Application Deployment	Installed on the customer's own hardware	Installed on vendor's hardware – delivered via the Internet	Installed on vendor's hardware – delivered via the Internet
Implementation	Lengthy implementation time	Lengthy implementation time	Faster implementation timeframes
Customization	Possible but expensive and time-consuming	Limited customization – tends to be very expensive	Customization limited but an inherent part of the application
Application Design	Monolithic client/server designs	Monolithic with a simple HTML interface	Designed for the Web environment and generally "lighter weight"
Upgrades	12+ months	Depends on developer's upgrade cycle	Generally monthly or even more frequently
Integration	Can be difficult and expensive	Can be difficult and expensive	Readily available via application programming interfaces (APIs)
IT Support	Generally provided by the customer	Generally provided by the customer, with some vendor support	Generally included in the package from vendor
Multi-Tenancy	Not applicable	Each customer requires an individual instance	Applications are designed to be multi-tenant
Hardware Requirements	Requires a specific operating system environment	Can require specific applications and operating environments	Delivered via a Web browser so generally operating system- and browser-agnostic
Accountability	Generally single-vendor so support path is well-defined	Hosting provider and software developer are two different organizations so accountability is complex	One vendor provides end-to-end solution so accountability is inherent

© Copyright 2010 - Diversity Analysis

An Overview of the Three Major Software Delivery Models

Perhaps one of the most important and fundamental steps we can take to understand the difference between the three major software delivery models is to properly define each model. Only then can we trace their evolution and understand their merits and drawbacks.

On-Premises Software Deployment

Before the widespread availability, affordability and adoption of networks – particularly the Internet – on-premises software deployments were virtually the only choice for businesses. Some examples include large-scale enterprise resource planning (ERP) and customer relationship management (CRM) systems as well as single-user programs such as QuickBooks or Microsoft Office. As its name implies, an on-premises implementation is software that's installed and operated on computer(s) located on the premises of the software licensee, rather than at a remote facility. This model largely defined and drove the first generation of business computing.

However, on-premises software is limited in its ability to support remote access to computing services. Customizations – if allowed – can be difficult and expensive. Software vendors also make significant investments in legacy code that tend to work poorly in off-premises configurations.

© Copyright 2010 - Diversity Analysis

Hosted Software Deployment

The increased penetration of the Internet proved to be an unstoppable influence on business software. In the 1990s, the combined forces of IT outsourcing and widespread Internet adoption led many vendors to develop on-premises-style commercial software – but host it off-site and license it to many different customers.

In a hosted deployment, the software physically resides at a third-party data center – the hosting site – where it is operationally managed by a service provider on behalf of the licensee. One off-shoot of this approach is the application service provider (ASP) model where the vendor provides additional domain expertise. ASPs specialize in software categories or even specific software vendors, helping to customize implementations for each customer.

Regardless of whether it is through an operational hosting provider or ASP, users access these remote computing resources over a network, typically the Internet. Each client company accesses its own dedicated instance of the application that is running at the remote data center – which creates significant overhead costs.

The compelling value proposition is that the cost to own and operate the software is far less if entrusted to an expert organization with the infrastructure and expertise solely dedicated to delivering that resource to the customer. Much of the IT overhead cost is transferred to the hosting provider – no longer does the organization need to build, staff, or maintain its own data center or support desk.

Unfortunately, much of the optimism surrounding the hosted software-delivery model was unfounded. Flawed business models – particularly for ASPs - combined with integration challenges have left this market segment in a state of stagnation. Many hosting providers are not the actual developers of the software program, leaving them largely at the mercy of the developer in terms of development timetables/roadmaps and features. What's more, most

© Copyright 2010 - Diversity Analysis

hosting providers are typically better-versed in hardware and networking issues and unused to dealing with software delivery and customization as well as end-user customer support challenges.

Software as a Service (SaaS)

Aiming to combine the best characteristics of both on-premises software and outsourced hosting, the Software as a Service (SaaS) model (sometimes referred to as "cloud computing") has emerged as an increasingly compelling alternative for many licensees. With SaaS, the application developer delivers shared, scalable services that the customer accesses over the Internet using a Web browser or mobile device. There's no need for the customer to own, license, understand, manage, or control the underlying hardware, software, or data/networking infrastructure that supports the service. SaaS is typically delivered on a term-based subscription basis, eliminating the need for upfront software licensing fees or major hardware purchases. The result is cost-effective, anytime/anywhere managed access that's increasingly attracting a growing number of enterprises and small/medium-sized businesses.

One of the key tenets of SaaS applications is a concept called "multi-tenancy." In SaaS deployments – all customers simultaneously access the same single instance of the application in the remote data center. This means the vendor can amortize a world-class infrastructure across multiple customers while reducing costs.

What's more, well-designed SaaS applications allow for instant provisioning and upward/downward scalability, parameter-driven customization and configuration and regular upgrades (instead of the daunting customization-breaking upgrades required with on-premises or hosted software implementations). So which model is best for your organization? As you conduct your evaluation, consider the following metrics and factors.

© Copyright 2010 - Diversity Analysis

The Pros and Cons at a Glance

On-premises Software

Pros:

- Greater end-to-end control of your software
- Maintain IP within the organization
- Implement significant customization

Cons:

- Generally higher cost
- Requires you to build and maintain your own IT infrastructure
- Must provide technical support, upgrades, and version control
- Upgrades may break any previous customizations
- Integration with third-party applications can be difficult/expensive

Hosted/ASP Software

Pros:

- Pushes infrastructure costs to the provider
- Potentially lower cost than on-premises deployment

Cons:

- Very limited customization
- Upgrades may break any previous customizations
- Support and service is typically an expensive extra charge
- Integration with third-party applications can be difficult/expensive
- Single tenancy limits the cost-reduction opportunities
- Customers have some security concerns
- Generally poor usability

Software as a Service

Pros:

- Core architecture makes it the most economical choice
- Ability to provision almost instantly
- Pushes infrastructure, service, and support costs to the vendor
- Integration is quick and easy using application programming interfaces (APIs)
- Customization available
- Upgrades are automatically applied to all users - all customers work with the same application
- Encourages alignment of people/process/system

Cons:

- Customers have some security concerns

© Copyright 2010 - Diversity Analysis

IT and Operational Requirements

Some of the more important considerations when choosing a software deployment model involve the software and hardware required and the people and skill sets required to properly operate and maintain that IT infrastructure. The following issues are key to any decision.

Software Procurement

Simply put, who procures the product and in what manner? With on-premises or hosted software, you will need to enter into a termed license and also purchase a maintenance contract simply to take physical possession of the software. And with on-premises software, the IT department must have a leadership role in the evaluation/selection process because they will have to manage the software as well as that software's potential impact on the rest of the corporate IT portfolio. By contrast, with SaaS, the business user can drive the procurement process – consulting appropriately with IT colleagues. What's more, since SaaS is procured through a monthly, quarterly or annual subscription fee, there are no upfront licensing fees (which often reach into six or seven figures) and there's a lower approval threshold and risk profile.

© Copyright 2010 - Diversity Analysis

Implementation

Who is responsible for implementing the chosen solution? For on-premises deployments, many companies find that they must implement the software themselves (a process that can take months – and sometimes years) or pay large dollars for expert consultants and implementation partners. Hosting providers handle much of the implementation headaches – they're experts, after all – but they are still deploying a dedicated (sometimes customized) instance of the software, which can take almost as long as on-premises software deployments. For SaaS implementations, either the vendor or one of the vendor's pre-approved implementation partners is responsible for rapidly provisioning access to the online application. Even with customizations, complex SaaS implementations can be completed in just days or weeks.

Naturally, a lengthy implementation cycle that stretches for many months means there's a far greater financial and operational risk to the customer. From software licensing to hardware to labor and consulting fees, significant costs are incurred from day one. But process and productivity improvements – and the payback they represent – can often be elusive, stretching years out into the future. By contrast, SaaS subscriptions require no upfront licensing fees and no lengthy implementation cycles and no attendant costs, translating into a far lower level of project risk.

© Copyright 2010 - Diversity Analysis

IT Operations and Management

Today's enterprise IT staff is already bearing a significant burden, so any new addition to the application portfolio must survive intense scrutiny and analysis before winning their approval. Major enterprise applications entail procurement, deployment, operations and maintenance cycles for software and IT infrastructure – tasks that fall to the licensee in the on-premises model or the hosting provider in the dedicated-instance hosted-software model.

With SaaS, those costs shift to the commercial software developer. The customer need only sign a subscription contract and the application service is accessible in much the same manner as a basic utility service: on-demand and scalable as needed. SaaS vendors achieve this by sharing their world-class infrastructure across thousands of customers. What's more, as the developers and full-time experts in providing this application service, the SaaS vendor is uniquely qualified to optimize the user's experience with respect to performance, availability and security. There's no internal IT learning curve to climb.

© Copyright 2010 - Diversity Analysis

Security

The ability of on-premises deployments to provide complete control over the application and infrastructure is initially quite appealing. Many organizations have resisted hosted or SaaS models because they don't feel comfortable storing and managing sensitive data in locations not under their direct control. However, putting security responsibilities on an internal team provides drawbacks. For example, the cost of proper information security measures can be prohibitive. Furthermore, on-premises customers must rely on the software vendor to release appropriate security patches to close any vulnerabilities in the application that may emerge.

With third-party application hosting, security is outsourced, cutting costs considerably. However, the effectiveness of that security depends on the size and capabilities of the service provider. The single-tenant architecture localizes any attack to a single customer and reduces the chances of any security attack affecting all customers. However, this can lead to complacency and move focus away from the security of their service. As with on-premises deployments, third-party hosters also passively rely on the software provider for security patches, opening them to the same disadvantages.

SaaS skeptics suggest that a lack of direct control over SaaS applications and data could result in a security nightmare for a business. However, this is a myth based on the erroneous assumption that SaaS providers cannot employ security professionals with the same strong expertise that many enterprises do. The reality is, a security breach for a SaaS company would affect all of its customers, which provides an incredible incentive to provide ironclad security measures. Their business is predicated on the ability to secure their infrastructure, applications data and operations. What's more, the SaaS model has a distinct advantage over on-premises and outsourced hosting models when it comes to patching application vulnerabilities. Since the SaaS provider is also the application developer, it can instantly patch application vulnerabilities.

© Copyright 2010 - Diversity Analysis

Upgrades

As any enterprise IT professional will tell you, upgrades are a never-ending nightmare in many organizations today. Just deploying the application (or applying a previous upgrade) can take months. In many instances, simply achieving a stable, reliable production environment is an extremely challenging task. And when you're done? Time to do it all over again.

On-premises software means internal enterprise IT teams can be consumed with an endless cycle of upgrades to business critical applications. What's more, the lengthy timeframes to upgrade can jeopardize service contracts or potentially introduce instabilities and incompatibilities with other operating systems, peripherals, applications and tools. The alternative, of course, is to hire an expensive team of consultants to hopefully expedite the process and lower the risk. For third-party application hosters, experience can be a major advantage. After upgrading a customer for the 20th time, they're likely very skilled. However, that takes time and consumes lots of resources that customers may need for other matters. What's more, upgrades can wreak havoc on both on-premises and hosted software deployments. New versions often "break" previously developed/applied customizations to commercial code.

By contrast, SaaS offers significant advantages in the upgrade process. First and foremost: there are no significant tasks for the customer. Some well-communicated advance notice likely mitigates any issues of disruption or preparation. The SaaS vendor is the application developer, so there are no learning curve issues. And in a multi-tenancy model, the upgrade only needs to be applied once to reach all customers. There's no need to "wait your turn" while waiting for a hosting provider to wend its way through a backlog of customers seeking the upgrade. What's more, since all customizations are already built into the commercial code, the upgrade has no disruptive impact on the production system. For a SaaS customer, an upgrade is a non-event, simply appearing one day without any intervention required.

© Copyright 2010 - Diversity Analysis

Business Continuity

As the operator of the software, the on-premises IT team or outsourced hosting provider must perform all of the routine – but critical – backup and off-site data storage tasks to ensure business continuity in the event of natural disaster, fire, civil unrest, or other interruption. There's no substitute for the business continuity plan and omitting it is not an option. And that means: wasted time, expense and overhead devoted to non-strategic activities – especially for on-premises deployments.

By contrast, a SaaS implementation alleviates the need for separate business continuity efforts related to that SaaS-deployed application. As part of the regular application subscription, the vendor backs up all customer data (according to pre-defined service level agreement parameters) off-site and deploys separate data centers and plans to provide uninterrupted service during any potential disasters or major unplanned downtime events.

© Copyright 2010 - Diversity Analysis

Ease of Customization

Regardless of deployment model, when a software solution doesn't meet all your needs out of the box, an important factor is how difficult is it to configure, customize, or tailor the application to your unique requirements? On-premises software offers excellent opportunities to build a customized solution and provides the highest level of customer control. However, highly customizable environments are expensive, time-consuming and not without risks. For instance, you might need to make changes to database tables, write new code in Visual Basic, or spread your customizations into many different areas of commercial code that was written by the vendor. (Of course, for lower-end commercial software, such as QuickBooks, there may not be any customization opportunities at all.)

What's more, when commercial upgrades are applied, your expensive customizations may "break" and require lengthy rework and retrofitting. Regression testing of your customizations – if it's done at all – might overlook subtle issues, incompatibilities and security vulnerabilities that you've unwittingly introduced. It's a similar story with third-party-hosted, dedicated instances of the application. One of the major shortcomings of outsourced application hosting is the lack of domain expertise, making integration with other systems a major headache.

One of the biggest myths about SaaS is that it lacks customization and integration options. Unlike what naysayers want you to believe, SaaS applications are rich with customization and integration features that are easy to access and configure. Although most SaaS applications offer customization from a branding point of view, many enterprise-level SaaS applications are now also offering the option to customize based on functional needs. The newest generation of SaaS applications actually offer customization capabilities that exceed what's possible with traditional on-premises software. Most importantly, SaaS customizations are preserved through the upgrade process, so the upgrade becomes a value-added non-event for the customer.

© Copyright 2010 - Diversity Analysis

Service and Support

The issues of service and support are somewhat easier to overlook, but their importance cannot be overstated. When you operate an on-premises software solution, you assume significant burdens and costs. You need to expand your existing (or even create a new) support desk for the application service. That means recruiting, training and retaining talented experts and delivering support and coverage as business needs dictate to support users.

There are IT-facing issues as well. The on-premises customer must ensure the infrastructure devoted to the application is running at peak performance with acceptable levels of reliability, availability, security and performance. Then, of course, when issues arise, it can be daunting to track down the source of the issue and hold the appropriate vendor accountable. Get ready for a lot of finger-pointing. Collectively, that's a tall order, which is why many outsource these tasks to hosting providers. However, many hosted-application arrangements are characterized by confusion over who provides the support: the hosting provider, the application developer, or the customer. In some models, the hosting provider provides direct support to end users. In other instances, the hosting provider only trains the customer's IT help desk, which then directly handles level-one support calls from users. Clarity on these matters is essential.

In the SaaS application-delivery model, it's simple. There is a single vendor providing the entire service who is responsible and accountable for both the product and service/support issues.

© Copyright 2010 - Diversity Analysis

Total Cost of Ownership

Any comparison of software deployment models must address the critical issue of cost. In this paper, we approach this from a total cost of ownership (TCO) perspective that encompasses everything from licensing or subscription fees to labor costs and infrastructure investments.

Even before the current economic downturn, businesses scrambled to reduce their operational costs. Not only are they looking to cut overall costs, but many are seeking to move costs out of capital budgets and into operating budgets. The drivers for this are simple - capital expenditures tend to have a long and complex approval process and create a disconnect between the revenue generated by a financial outlay, and the outlay itself. Moving expenditures to the operating line more closely aligns them with revenue generation, eases the approvals process and avoids the "sunk cost" issue where organizations are reluctant to move away from projects with significant historical capital expenditure - regardless of their effectiveness.

On-premises is the most expensive of the three models, presenting a very high TCO. The infrastructure costs for running the data center, licensing fees, upgrade and maintenance fees and labor costs result in extensive capital and operating expenses. The restrictions and barriers presented by traditional software applications combined with the responsibilities arising from deployment and management of the applications inside the data center often makes on-premises applications prohibitively expensive with uncertain ROI and very long payback periods. Other factors contributing to the runaway TCO of on-premises applications include implementation costs and internal support costs. In a challenging economic climate, the on-premises model may well turn out to be a business handicap for any organization.

© Copyright 2010 - Diversity Analysis

Although outsourced hosting is less expensive than the on-premises model, it still presents higher costs that SaaS because of the single tenancy delivery model and the significant licensing costs. There are other components to TCO like the cost of application upgrades, additional support costs and more. Compared to the SaaS model, the outsourced hosting model has proven to be very cost inefficient.

SaaS is the most cost-effective model with a very low TCO. First, the SaaS vendor assumes more responsibility than hosting service providers - demanding no capital expenditures from the customer. And since the SaaS vendor is also the actual application developer, licensing costs are also not a factor. SaaS vendors utilize a multi-tenant architecture to save on the costs of offering the application as a service. This cost savings is passed on to customers resulting in a lower TCO. What's more, the absence of any upgrade, implementation, or maintenance fees accelerates the payback of the SaaS approach. The cost components of SaaS are for subscriptions and training. Compared to the huge upfront licensing and training fees for traditional on-premises software, the cost of SaaS is much lower.

© Copyright 2010 - Diversity Analysis

Conclusion: Software Orientation, Service Orientation

Given the numerous operational, functional and financial advantages, SaaS deployments are gaining increased attention from enterprises seeking smarter, more efficient ways to access and deploy critical computing resources inside and outside their organizations. With its software orientation and service commitment, SaaS in many ways represents a "best of both worlds" perspective, combining the control and innovation of on-premises software with the speed and convenience of the outsourced hosting models – all with rapid innovation and lower cost. As this innovative software-delivery model gains further momentum, software evaluators would be well-advised to give SaaS careful consideration.

© Copyright 2010 - Diversity Analysis

About Intacct

Intacct is the market and technology leader in cloud computing financial management and accounting applications for small and midsized businesses, and the preferred financial applications for AICPA business solutions. Intacct applications are used by thousands of businesses from startups to public companies and are designed to improve company performance and make finance more productive. The Intacct system includes project accounting, contract management, revenue recognition, inventory, purchasing, vendor management, financial consolidation and financial reporting applications, all delivered over the Internet via Software as a Service (SaaS). Intacct is headquartered in San Jose, Calif. For more information, please visit www.intacct.com or call 877-437-7765.

© Copyright 2010 - Diversity Analysis

About Diversity Analysis

Diversity Analysis is a broad spectrum consultancy specialising in SaaS, Cloud Computing and business strategy. Our research focuses on the trends in these areas with greater emphasis on technology, business strategies, mergers and acquisitions. The extensive experience of our analysts in the field and our closer interactions with both vendors and users of these technologies puts us in an unique position to understand their perspectives perfectly and, also, to offer our analysis to match their needs. Our Analysts take a deep dive into the latest technological developments in the above mentioned areas. This, in turn, helps our clients stay ahead of the competition by taking advantage of these newer technologies and, also, by understanding any pitfalls they have to avoid.

Our Offerings: We offer both analysis and consultancy in the areas related to SaaS and Cloud Computing. Our focus is on technology, business strategy, mergers and acquisitions. Our methodology is structured as follows:

- Research Alerts
- Research Briefings
- Whitepapers
- Case Studies

We also participate in various conferences and are available for vendor briefings through Telephone and/or Voice Over IP.

© Copyright 2010 - Diversity Analysis

About the Authors

Ben Kepes

Ben is the founder and managing director of Diversity Limited, a consultancy specializing in Cloud Computing/SaaS, Collaboration, Business strategy and user-centric design. He is also editor of Diversity Blog, a specialist Cloud Computing blog. More information on Ben and Diversity Limited can be found at http://diversityanalysis.com/

Krishnan Subramanian

Krish is an entrepreneur, open source researcher and evangelist, cloud computing analyst and consultant, ex-physicist, and one of the industry's most prolific bloggers. He is part of the management team in two companies in India and sits on the board of a few privately owned companies. More information about Krish can be found at http://www.krishworld.com/.

© Copyright 2010 - Diversity Analysis