From: Boone, Cory

Sent: 6/23/2014 11:53:44 AM

To: TTAB EFiling

CC:

Subject: U.S. TRADEMARK APPLICATION NO. 85979129 - DMX MOBILE - DMX Mobile - SU - Request for Reconsideration Denied - Return to TTAB - Message 2 of 5

****

Attachment Information:

Count: 3

Files: soft5-3.jpg, soft5-4.jpg, soft5-5.jpg

Unlike traditional software which is conventionally sold as a perpetual license with an up-front cost (and an optional ongoing support fee), SaaS providers generally price applications using a subscription fee, most commonly a monthly fee or an annual fee.[14] Consequently, the initial setup cost for SaaS is typically lower than the equivalent enterprise software. SaaS vendors typically price their applications based on some usage parameters, such as the number of users using the application. However, because in a SaaS environment customers' data reside with the SaaS vendor, opportunities also exist to charge per transaction, event, or other unit of value.

The relatively low cost for user provisioning (i.e., setting up a new customer) in a multi-tenant environment enables some SaaS vendors to offer applications using the freemium model. In this model, a free service is made available with limited functionality or scope, and fees are charged for enhanced functionality or larger scope. Some other SaaS applications are completely free to users, with revenue being derived from alternate sources such as advertising.

A key driver of SaaS growth is SaaS vendors' ability to provide a price that is competitive with on-premises software. This is consistent with the traditional rationale for outsourcing IT systems, which involves applying economies of scale to application operation, i.e., an outside service provider may be able to offer better, cheaper, more reliable applications.

Architecture [edit]

The vast majority of SaaS solutions are based on a multi-tenant architecture. With this model, a single version of the application, with a single configuration (hardware, network, operating system), is used for all customers ("tenants"). To support scalability, the application is installed on multiple machines (called horizontal scaling). In some cases, a second version of the application is set up to offer a select group of customers with access to pre-release versions of the applications (e.g., a beta version) for testing purposes. This is contrasted with traditional software, where multiple physical copies of the software — each potentially of a different version, with a potentially different configuration, and often customized — are installed across various customer sites.

While an exception rather than the norm, some SaaS solutions do not use multi-tenancy, or use other mechanisms—such as virtualization—to cost-effectively manage a large number of customers in place of multi-tenancy.[15] Whether multi-tenancy is a necessary component for software-as-a-service is a topic of controversy.[16]

Characteristics [edit]

While not all software-as-a-service applications share all traits, the characteristics below are common among many SaaS applications:

Configuration and customization [edit]

SaaS applications similarly support what is traditionally known as application *customization*. In other words, like traditional enterprise software, a single customer can alter the set of configuration options (a.k.a., parameters) that affect its functionality and look-and-feel. Each customer may have its own settings (or: parameter values) for the configuration options. The application can be customized to the degree it was designed for based on a set of predefined configuration options.

For example: to support customers' common need to change an application's look-and-feel so that the application appears to be having the customer's brand (or—if so desired—co-branded), many SaaS applications let customers provide (through a self service interface or by working with application provider staff) a custom logo and sometimes a set of custom colors. The customer cannot, however, change the page layout unless such an option was designed for.

Accelerated feature delivery [edit]

SaaS applications are often updated more frequently than traditional software,[17] in many cases on a weekly or monthly basis. This is enabled by several factors:

- ***The application is hosted centrally, so an update is decided and executed by the provider, not by customers.***
- The application only has a single configuration, making development testing faster.
- The application vendor has access to all customer data, expediting design and regression testing.
- The solution provider has access to user behavior within the application (usually via web analytics), making it easier to identify areas worthy of improvement.

Accelerated feature delivery is further enabled by agile software development methodologies.[18] Such methodologies, which have evolved in the mid-1990s, provide a set of software development tools and practices to support frequent software releases.

Open integration protocols [edit]

Since SaaS applications cannot access a company's internal systems (databases or internal services), they predominantly offer integration protocols and application programming interfaces (APIs) that operate over a wide area network. Typically, these are protocols based on HTTP, REST, SOAP and JSON.

The ubiquity of SaaS applications and other Internet services and the standardization of their API technology has spawned development of mashups, which are lightweight applications that combine data, presentation and functionality from multiple services, creating a compound service. Mashups further differentiate SaaS applications from on-premises software as the latter cannot be easily integrated outside a company's firewall.

Collaborative (and "social") functionality [edit]

Inspired by the success of online social networks and other so-called *web 2.0* functionality, many SaaS applications offer features that let its users collaborate and share information.

For example, many project management applications delivered in the SaaS model offer—in addition to traditional project planning functionality—collaboration features letting users comment on tasks and plans and share documents within and outside an organization. Several other SaaS applications let users vote on and offer new feature ideas.

While some collaboration-related functionality is also integrated into on-premises software, (implicit or explicit) collaboration between users or different customers is only possible with centrally hosted software.

Adoption drivers [edit]

Several important changes to the software market and technology landscape have facilitated acceptance and growth of SaaS solutions:

- The growing use of web-based user interfaces by applications, along with the proliferation of associated practices (e.g., web design), continuously decreased the need for traditional client-server applications. Consequently, traditional software vendor's investment in software based on fat clients has become a disadvantage (mandating ongoing support), opening the door for new software vendors offering a user experience perceived as more "modern".
- The standardization of web page technologies (HTML, JavaScript, CSS), the increasing popularity of web development as a practice, and the introduction and ubiquity of web application frameworks like Ruby on Rails or languages like PHP gradually reduced the cost of developing new SaaS solutions, and enabled new solution providers to come up with competitive solutions, challenging traditional vendors.
- The increasing penetration of broadband Internet access enabled remote centrally hosted applications to offer speed comparable to on-premises software.
- The standardization of the HTTPS protocol as part of the web stack provided universally available lightweight security that is sufficient for most everyday applications.
- The introduction and wide acceptance of lightweight integration protocols such as REST and SOAP enabled affordable integration between SaaS applications (residing in the cloud) with internal applications over wide area networks and with other SaaS applications.

Adoption challenges [edit]

Some limitations slow down the acceptance of SaaS and prohibit it from being used in some cases:

- Since data are being stored on the vendor's servers, data security becomes an issue.[19]
- SaaS applications are hosted in the cloud, far away from the application users. This introduces latency into the environment; so, for example, the SaaS model is not suitable for applications that demand response times in the milliseconds.
- Multi-tenant architectures, which drive cost efficiency for SaaS solution providers, limit customization of applications for large clients, inhibiting such applications from being used in scenarios (applicable mostly to large enterprises) for which such customization is necessary.
- Some business applications require access to or integration with customer's current data. When such data are large in volume or sensitive (e.g., end users' personal information), integrating them with remotely hosted software can be costly or risky, or can conflict with data governance regulations.
- Constitutional search/seizure warrant laws do not protect all forms of SaaS dynamically stored data. The end result is that a link is added to the chain of security where access to the data, and, by extension, misuse of these data, are limited only by the assumed honesty of 3rd parties or government agencies able to access the data on their own recognizance.[20][21][22][23]
- Switching SaaS vendors may involve the slow and difficult task of transferring very large data files over the Internet.
- Organizations that adopt SaaS may find they are forced into adopting new versions, which might result in unforeseen training costs or an increase in probability that a user might make an error.
- Relying on an Internet connection means that data are transferred to and from a SaaS firm at Internet speeds, rather than the potentially higher speeds of a firm's internal network.[24]

The standard model also has limitations:

- Compatibility with hardware, other software, and operating systems.[25]
- Licensing and compliance problems (unauthorized copies of the software program putting the organization at risk of fines or litigation)
- Maintenance, support, and patch revision processes.

Emerging trends [edit]

As a result of widespread fragmentation in the SaaS provider space, there is an emerging trend towards the development of SaaS Integration Platforms (SIP).[26] These SIPs allow subscribers to access multiple SaaS applications through a common platform. They also offer new application developers an opportunity to quickly develop and deploy new applications. This trend is being referred to as the "third wave" in software adoption - where SaaS moves beyond standalone applications to become a comprehensive platform. Zoho and Sutisoft are two companies that offer comprehensive SIPs today. Several other industry players, including Salesforce, Microsoft, and Oracle are aggressively developing similar integration platforms.

Data escrow [edit]

Software as a service data escrow is the process of keeping a copy of critical software-as-a-service application data with an independent third party. Similar to source code escrow, where critical software source code is stored with an independent third party, SaaS data escrow is the same logic applied to the data within a SaaS application. It allows companies to protect and insure all the data that reside within SaaS applications, protecting against data loss.[27]