OpSource™
The Business of Web Operations

# Transforming Your Software Product Into a Service

**WHITE PAPER**

By Accelerance & OpSource

## Introduction

There is growing market demand for more economical and efficient enterprise applications to an ever-expanding global market. The combination of the ubiquitous Internet and the availability and legitimacy of open source software is creating substantial opportunities and economies for software vendors to deliver Software as a Service (SaaS).

Software as a Service is a model in which the software vendor provides an Internet hosted version of their application (in house or at a managed 3rd party site) that is accessed by customers from the website and paid for on a per-use, per-project or subscription basis. Salesforce.com is a leading example of the SaaS model.

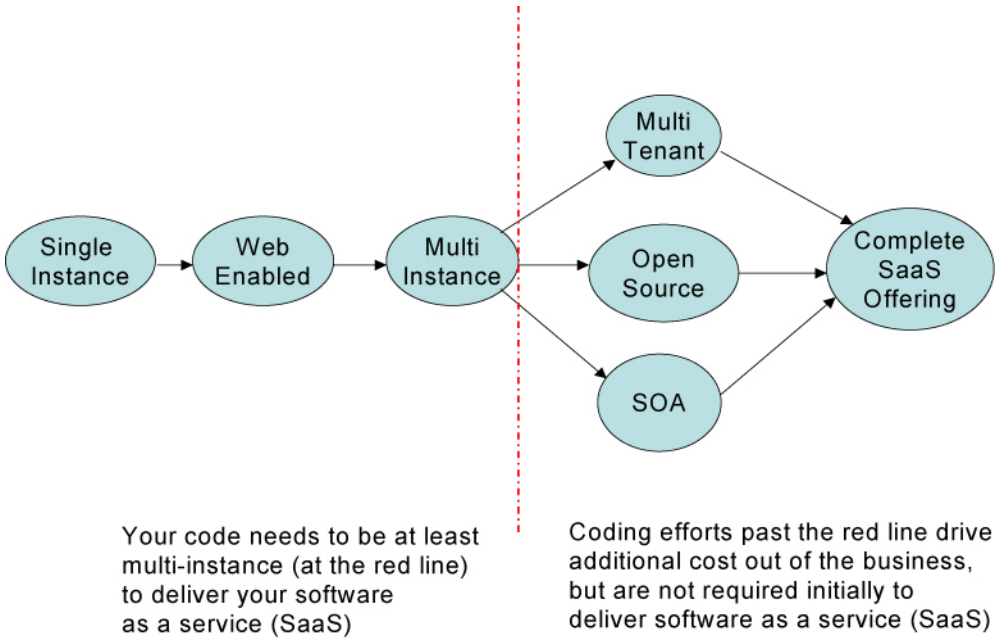
The SaaS model offers significant benefits to software vendors and their customers. The SaaS model offers customers cost-effective subscription-based or per-use pricing, eliminating the need for substantial capital outlays to purchase perpetual software licenses. It also eliminates the initial outlay and on-going costs and risks of installing, supporting and maintaining in-house hardware and the associated IT staff. In addition, user access and application performance can be dramatically improved with Internet-based, on-demand, 24x7 systems. The SaaS model opens new markets to software vendors. Established software companies can broaden their market reach by offering SaaS solutions to small and midsized enterprises. Other benefits include the financial advantages of predictable recurring revenue streams and strengthened relationships with customers. Software vendors migrating to or developing products from the outset as SaaS offerings will have a significant competitive advantage when competing with traditional license-model vendors.

Realizing the benefits of the SaaS model may require fundamental changes to a software vendor's business model, software architecture and operational structure. This white paper provides an overview of the issues associated with the software application itself and the development considerations associated with moving to a SaaS model.

Time is of the essence. As with any new business model, the rewards often go to early market entrants. Salesforce.com has broken down the barriers to the SaaS model, and many others are following. Accelerating time-to-market is critical to your business success. Outsourcing product modifications to implement your SaaS offering, with the assistance of an expert services team, and engaging an optimal on-demand service delivery firm will accelerate your time-to-market and insure an on-time, on-budget, on-scope implementation.

## The Challenge of Transforming Your Software

While there are a multitude of benefits in providing Software as a Service, traditional software companies may face challenges in moving to this model. First, your software must be web-enabled with all functions carried out by the user using a web browser. If you have a client-server application, you must replace the functionality implemented in the client with HTML, and possibly other technologies (XML, Java, etc.), that can be displayed by a web browser over the Internet. Next, in order to gain operational efficiency, your software needs to be multi-instance. You move from single-instance to multi-instance by loading multiple copies of your software on a single set of servers. Multi-instance enables you to share the cost of a server across multiple customers. This will move you to the red line in the following diagram and allow you to offer SaaS. Additional productivity enhancements and economies may be gained by moving to multi-tenant SaaS, or replacing proprietary commercial software with open source software. Web services provide an opportunity for integration with other applications and data flows.




**Single Instance Applications**

Traditional client/server applications are single instance. They require software to be installed on the user's computer to carry out computations and provide functionality. Clients often implement highly interactive features and enable the user to manipulate large amounts of data. This can be very difficult to implement in a traditional HTML, request/reply web application interface that requires frequent page refreshes. Migrating from client/server to an Internet-based SaaS model is highly dependent on your specific application.

Today, new Rich Internet Application (RIA) technology is available from Macromedia, Laszlo Systems and others that give web applications the look-and-feel and functionality of a desktop application or client. RIA requires little or no software be installed on the user's client computer. The most that is needed is a small browser plug-in or Java applet. This fundamental change to the user interface converts your client /server application to a single-tenant web application.

Web applications may be single instance or multi-instance. A single-instance web application is typically installed on dedicated servers in the customer's data center and used only internally, behind the firewall. At installation time, your software is configured to consume whatever system resources are needed and available on the computers.

When a web application is offered as a service over the Internet, it should be hosted in a professional data center. This will minimize costs and delivery high quality service to your customers. If you have a single instance application and more than one customer, one approach is to install a new instance of your software on a dedicated server for each customer. This may work for a few customers or some big accounts, but it does not scale effectively for large numbers of customers. It also cannot be used for small and medium sized customers that cannot afford the set-up costs.

**Moving from Single to Multiple Instances**

An alternative to individual customer dedicated servers is to install multiple copies of your software on a single set of servers. This is called multi-instance. Multi-instance enables you to share the cost of a server across multiple customers. Most business applications use a database and each additional copy of the software installed requires a new database instance as well.

Installing multiple copies of your software on one set of servers may not be as easy as it sounds. Installation procedures need to be modified so that each instance is installed without disrupting resource allocation or the security of the other previously installed copies of the software.

There is a limit to the number of instances that can be installed and eventually system resources will be consumed. System resources include shared memory, process semaphores and other internal operating system parameters. So the question becomes, “How many copies of your software can you install on a server?”

Obviously, you can keep installing instances of your software until resources are exhausted. However, you must also consider the performance of the system under load by users. Typically there are a maximum number of simultaneous users your software must support and minimum performance or response time requirements that must be met to satisfy customer commitments.

An accurate answer to the “How many copies of your software can you install on a server?” question is derived by testing the software as you add additional instances. This is best done with automated testing software tools that can simulate the desired number of users placing a load on the system.

The testing process is to determine the optimal number of instances and the resulting performance. This is accomplished by installing additional instances of your application, and carefully monitoring system resources and running user load tests using variable traffic modeling to determine the point at which returns diminish.

This process of maximizing the number of instances on the servers can take one to three weeks depending on the size and complexity of your system, the quality of your installation process whether you have already created automated user load testing scripts and procedures.

Minor code changes may be needed to move to multi-instance. For example, if your application reads and writes a file with a hard-coded filename and location on the disk, then the file must be created in different locations for each instance to avoid conflicts between each instance. These problems will be discovered and changes will need to be made during the one to three weeks.

**Next Steps - Improving Functionality and Reducing Costs**

Once your software is running effectively as a multi-instance SaaS application, you may want to pursue a multi-tenant architecture. In a single instance, multi-tenant architecture, multiple customers share a single instance of your software. Migration from multi-instance to multi-tenant can be a significant project and may even require a rewrite of your application from the ground up. The efficiencies gained in moving to multi-tenancy need to be closely examined. You might find your resources better spent in other ways.

**Another possible step would be to focus on driving costs out of your model. Many applications have dependencies on expensive proprietary databases and/or middleware. Significant savings can be realized by migrating to lower cost or open source alternatives. An investment here might provide significant savings in operating costs that would be transparent to your end users and very beneficial to your bottom line.**

You might also consider adding web services for inter-process communications. This will be particularly appealing if your application is part of a workflow with information passing-to or gathering-from another application. Designing with web services in mind will minimize long-term integration requirements.

**A Single Instance, Multi-Tenant Web Application**

Software companies have created web applications for over ten years now. These are often installed on the Intranet of a customer and only used internally, behind the firewall. This single instance of the software is used by just one customer. This is both single-instance and single-tenant.

You saw above how you can install and test your software to make it multi-instance – having multiple copies running on one server. However, each copy is a single-tenant web application.

Single-tenant web applications can be modified to support multiple customer tenants on the same instance. Multi-tenant web applications minimize the amount of hardware needed to support multiple customers. Also, customers can self-provision their use of your software by signing up for an account and entering payment information. This minimizes, and often eliminates, the amount of support needed to set up a new customer.

One of the modifications to support multi-tenant is the creation of a user interface for user provisioning of accounts in the system. Another modification, depending on the requirements for integration with other enterprise systems, is an LDAP interface for convenient provisioning and administering of user accounts. Modern database technology can enable quick duplication of the data model so each customer has its own copy of each table in the database. This is an elegant way to keep customer data separate when stored in the single database instance used for the service.

Templates for configuration of the software should be provided to accelerate customization and adoption of the service by new customers. Templates support various scenarios of system usage by customers.

A system management dashboard showing system use by all tenants may be required. A mechanism must be available to measure system usage for purposes of billing as well as monitoring system load. Administrative accounts for customer support purposes may also need to be implemented.

It may be necessary to enhance the reliability of the back-end, using database technology to implement parallel servers at physically distant locations, to ensure constant up time during periods of natural or man-made disasters.

**Maintaining Performance of Your Multi-Tenant Web Application**

Multi-tenant applications must deal with several issues that are not as pronounced in single-tenant and client/server systems. Because multi-tenant systems are available over the public Internet, usage may be unpredictable. Therefore, demand planning must be done more carefully. The systems should be instrumented to detect increasing usage so additional hardware and bandwidth are provided to maintain service levels.

**Driving Down Costs by Moving to Open Source**

Many software developers are agnostic about the application server and database software used by their applications. The customer often dictates these choices. If your customers want to use Oracle as the database, then you must support this popular choice. Your software must have modules to support each database technically. Business-wise, you pass along the cost of the database license to the end customer, if they do not already own a license.

But what database should you choose for your software when it is offered as a service? There may not be a need for the technical features of an expensive commercial database. Moreover, the economics of offering your software as a service may preclude the expense of a commercial database license fee.

Therefore, many companies converting their software to a service will choose one of the low or no cost open source databases available today. These database choices are now widely used and robust. Advanced features such as redundant clustering and automated backup capabilities rival those of commercial databases.

If your application does not yet support one of these databases, a few technical issues need to be overcome. The format and syntax of most SQL used to access and manipulate data in a database is standard. However, almost every database vendor extends SQL and many applications use these extensions, such as special functions to modify and compare data. There can be many variations in how each database vendor treats cursors, triggers, data types and package variables. If you use SQL extensions in your application, you will need to recode these SQL statements to work with the target open source database.

Migration to on demand delivery models works cohesively with bootstrapped technology deployment and investment. Even if the open source database software does not have all the features you want to have or if they run a little slower, you may have no choice economically when you first start offering your software as a service. It

may not make financial sense for you to invest tens of thousands of dollars in a commercial database license while you can only charge a few hundred dollars per subscriber. Over time, as your subscriber base grows, you may choose to switch to the commercial database. Until you can afford it or activity levels grow to high levels, open source database solutions may be your only practical solution.

| Application Servers | | Databases | |
|---|---|---|---|
| **Commercial** | **Open Source** | **Commercial** | **Open Source** |
| BEA WebLogic | JBoss | MS SQL Server | MySQL |
| IBM WebSphere | Tomcat | Oracle | PostgreSQL |
| Oracle App Server | JOnAS | Sybase | CloudScape |
| | Enhydra | DB2 | Firebird |
| | Geronimo | | |

Another relatively expensive part of your software is the license required for a commercial Java application server. This is another category of software where several open source options exist. Generally, conversion over to an open source application server is relatively straightforward. All must comply with the specification for Java 2 Enterprise Edition (J2EE) and your code should not need any modifications.

However, there are differences in how you install your code in the application server. The installation and set up process is well documented for all open source application servers. You must modify your installation process to accommodate the requirements of the application server you use.

Again, the business case is clear. A huge community of users has made open source application servers a safe choice. The cost of a commercial application server is difficult to justify when you are just starting out offering your software as a service. As with the conversion to-and-from an open source database, you can always switch back to a commercial application server as your subscriber base grows.

**Web Services For Data Transfer and Integration**

When customers install your software in their own data center, behind their firewall, they are able to integrate the software with other applications and data sources. When you make your software available as a service over the Internet, then integration is not as easy. Authentication and encryption must be provided to enable safe data transfers.

The most popular approach to data transfers and integration over the public Internet is with web services, the SOAP protocol and WSDL. If your application has an Application Programming Interface (API) in a native language like Java or C++, you will need to create a web services interface that uses the API to communicate with your software and enables bidirectional data flow with the external world using SOAP.

**Time to Market**

Time is of the essence. As the new SaaS model is adopted, early entrants will have a significant advantage. Evolving your application to web-enabled, multi-instance will allow you to become a SaaS player quickly. Time to market issues should be considered when deciding whether to partner with experts or pursue migration and infrastructure development in-house.

**The Advantages of Outsourcing Software Development**

Outsourced developers, who are experienced with SaaS, can help you move forward quickly in migrating to this new model. They can provide installation and load testing to determine the optimal set-up for your multi-instance configuration; adapt your software to migrate from multi-instance to multi-tenant; or develop a multi-instance or multi-tenant application from your client/server application.

Importantly, an outsourced developer can modify your existing software product without disrupting the flow of new features and enhancements that your present customers expect. With a managed outsourcing relationship, you can continue to focus on your current business while outsourced developers are creating software to support your new business model. Outsourced developers will provide you both a cost and time savings in reaching the SaaS model.

**Why Accelerance & OpSource**

To take the competitive lead in your space by leveraging SaaS, you should consider working with experienced vendors such as Accelerance and OpSource to guide you through the process of transforming your application.

Accelerance has the skills and experience to help you develop your SaaS software product. Accelerance is an expert outsourcing services firm working with small-to-medium sized businesses that are grappling with the issues of outsourcing software development. Accelerance provides its clients with the information and services they need to begin outsourcing quickly and with risk-free results. Accelerance's Vision services include: Vision Decision - assessment of the in-house versus outsourced development decision; Vision Resources - independent, global resource selection with 17 teams in 14 countries, employing over a thousand engineers; Vision Success - expert project execution; and Vision Perfect - quality assurance testing. Accelerance, located in Los Altos, California (the heart of the Silicon Valley), was founded in 2001 by Steve Mezak, a veteran of numerous software development companies and projects both onshore and offshore. For more information, visit: www.accelerance.com.

## About OpSource

OpSource™ delivers Software-as-a-Service (SaaS) and Web applications for on-demand companies, with hundreds of applications, millions of users and billions of transactions supported daily. OpSource On-Demand™, the leading Web operations solution, is defining how Web-based software is delivered. By choosing OpSource as their Web application delivery partner, companies are freed from investing in and managing the complex and costly infrastructure and services necessary to deliver applications over the Web. They can instead focus their resources on developing, marketing and selling their applications and services. Further, by using OpSource Connect™ companies can leverage Web services such as OpSource Billing CLM™, OpSource Analytics™ and OpSource End-User Support™ and integrate their applications with other SaaS applications over the Internet as well as with enterprise applications behind the corporate firewall. OpSource On-Demand is suitable for companies at any stage of growth, with any type of on-demand application. OpSource is the only company to offer Success-Based Pricing, a pricing model that allows businesses to begin with a modest minimum commitment and scale expenses as revenues increase.

Headquartered in Santa Clara, CA, OpSource has Web application delivery centers in Virginia, London and Bangalore. For more information about OpSource, visit www.opsource.net.

OpSource™
The Business of Web Operations

**Corporate Headquarters**
5201 Great America Parkway
Suite 120
Santa Clara, CA 95054
1-800-664-9973 (USA)
+44 207 043 1240 (UK)
sales@opsource.net
**www.opsource.net**