

Software as a Service

The Next Generation of Software Deployment

A VeraBridge white paper on how the "SaaS" model for software delivery benefits financial services organizations

EXECUTIVE SUMMARY

Executives are questioning traditional methods of paying for and deploying software

The traditional approach to software licensing and deployment is perceived as high cost, front-loaded with big license fees, high risk, and taking far too long to implement.

Software as a Service (SaaS) is emerging as a better approach to software deployment

SaaS is a software distribution model where applications are managed and hosted by a vendor and made available to clients via the internet on a subscription basis.

SaaS has significant business benefits

Lower costs, less risk, and shorter deployment are the main benefit areas; more specifically:

- ***No Upfront License Fee:*** SaaS-deployed software is subscription based, usually a monthly fee.
- ***Lower Total Cost of Ownership:*** Over three years, SaaS is 50% - 70% less than traditional software.
- ***Investment Aligned with Value with Variable Pricing:*** The fees for SaaS-deployed software are typically based on a usage metric (users, offices, accounts, etc.) which directly aligns costs with business value.
- ***No IT Staff Impact:*** There are no IT requirements for clients to support or operate the application.
- ***Maintain Control:*** Clients still have control of the rules, workflow, parameters, and processing in the application . . . without the burden of the back office infrastructure.
- ***Scalability:*** Companies can use more of the application as business needs evolve without adding infrastructure.
- ***Technology Access:*** As the SaaS-deployed applications evolve and incorporate new capabilities, companies have access to advancements without costs to implement upgrades.
- ***No Long-Term Commitment:*** Most SaaS providers require little, if any, minimum term, so customers use the software only as long as it adds value.

VeraBridge applications are delivered in a SaaS model

All VeraBridge applications for financial services are provided as SaaS applications. Monthly fees are based on usage levels, with no long-term commitment.

PROBLEMS CLIENTS HAVE WITH THE TRADITIONAL SOFTWARE BUSINESS MODEL

"I don't want to buy software . . . I just want what it does."

Head of operations, major international insurer

This senior executive was not being facetious. Code on a computer disk somewhere is not what executives want; they want what it does for their business; like increasing revenues, controlling expenses, improving customer service, rolling out new products, or any of hundreds of areas of potential business value.

But, getting the benefits means getting the software. In a traditional deployment model, the software is either developed by the IT staff and consultants, or purchased (more precisely, "licensed") from an applications vendor.

Business and IT executives are increasingly frustrated with the traditional software deployment model; issues that tend to fall into one of three categories.

1. Costs License: Software licenses involve a large upfront license investment, many times into seven figures.

Maintenance: Ongoing annual maintenance can be 15% - 25% of the license fee, meaning the every four to six years the software is "re-bought."

Implementation: Whether implemented by staff or consultants, the costs can be substantial to get from a signed license to "live" use of the software. While investment in implementing software varies considerably depending on the type of application, companies often use a rule of thumb of 3 – 5 times the license fee will be spent during implementation.

Infrastructure: Operating the software is a major expense as well. Obvious costs like hardware, training, and ancillary software are compounded by the need for additional IT staff. A study by Gartner Group estimates 70% of the five-year total cost of ownership is spent after implementation.

Customizations: Few organizations can use sophisticated software "out of the box." Customizations as simple as special reports to as complex as unique processing add substantially to the total cost of ownership. The costs are incurred to develop the customizations, then ongoing costs to "reapply" these customizations when new releases of the software are available.

2. Risks Software projects are very complex, and risk of failure is high. The financial industry is awash with stories of implementations that just crashed; millions spent with nothing ever implemented.

But it's not just implementation risk . . . there's risk that benefits promised by vendors are never realized. And, often these benefits, even if realized, are so far in the future they are hard to quantify.

3. Time Many executives could live with the huge costs if the software could be implemented in a reasonable period of time. There are too many instances of implementations taking 12 – 18 months or more. As one executive told us recently *"This has been going on so long I forgot why we were doing this!"*

Simply put, the traditional software model is too costly, too risky, and too long to get the value.

DIFFERENT APPROACHES HAVE BEEN TRIED

Since the first computer was used in business, there has been executive interest in *“letting someone else run the computer while we run the business.”* This interest has been driven by strategy, by culture, and by costs.

From at least the 1970’s, there have been alternative business models to help companies get value from software while mitigating some of the cost, risk, and time issues. These approaches have gone under a variety of names; *service bureau, outsourcing, hosted applications, on-demand software, and application service provider* to name a few. They tended to appeal to small and mid-size financial services companies that could not afford the hardware and support staff to operate the multitude of applications needed to run even a small bank or insurance company.

These approaches, however, were really no different than the traditional software model. In some cases, a license fee was replaced by a monthly fee, but the same cost, risk, and time issues were present. The only real difference is that the software was *“running on a box at the vendor site rather than the customer’s site.”*

SAAS IS THE NEW APPROACH

SaaS is a software distribution model where applications are managed and hosted by a vendor and made available to customers over the internet on a subscription basis.

Clients realize a number of benefits from SaaS:

- **No Upfront License Fee:** SaaS-deployed software is subscription based, usually a monthly fee.
- **Lower Total Cost of Ownership:** Over five years, SaaS is much less than traditional software models.
- **Variable Pricing Aligns Investment with Value:** The fees for SaaS-deployed software are typically based on some usage metric (users, offices, accounts, etc.) which more directly aligns costs with business value.
- **No IT Staff Impact:** There are no customer IT requirements to support or operate the application.
- **Scalability:** Clients use more of the application as business needs evolve without adding infrastructure.
- **Technology Access:** As the SaaS-deployed applications evolve and incorporate new capabilities, companies have access to advancements without additional costs.
- **No Long-Term Commitment:** Most SaaS providers require little, if any, minimum term, so customers use the software only as long as it adds value.

TYPICAL SAAS PRICING COMPONENTS

“If it appreciates, I want to own it. If it depreciates, I want to rent it. Technology only depreciates.”

Insurance executive explaining why SaaS model makes since to her.

In simplest terms, using a SaaS application involves “renting” the application, infrastructure, and support from the vendor for a “pay-as-you-go” monthly fee based on some level of usage.

SaaS pricing usually involves two components.

Implementation Costs

Depending on the complexity of the application, implementation can be trivial or somewhat involved. But, in virtually all cases SaaS-deployed applications implement much faster than equivalent onsite software.

For simple applications, implementation may involve nothing more than an online registration, then inputting whatever data is relevant. Salesforce.com is an example of this type of application.

More complex applications may involve what appears to be a more traditional software implementation (e.g., conversion of existing data, development of new models, etc.).

However, implementations tend to be faster with SaaS-deployed applications for two principal reasons:

1. No New Infrastructure: There is no need to set up infrastructure at the company’s site. This saves money, of course, but it also saves the time of requisitioning hardware, licensing ancillary software, getting internal approvals, etc. In large organizations, getting the infrastructure ready can take months.
2. Little or No Integration: Much of the work implementing traditional onsite software is spent integrating with other applications, databases, middle layers, etc. SaaS-deployed applications tend to be fully integrated from the user interface to the underlying database.

Usage Fee

Usage is the monthly fee for using the software. But, usage is much more than just a proxy for a license to the software. The monthly fee includes:

- Access to the Software: Analogous to a license to use the software.
 - Maintenance: Upgrades to the application.
 - Enhancements: Improvements to functionality for general release.
 - Hosting and Infrastructure: Including hardware and ancillary software to run the application.
 - Help Desk: Availability of specialists to support users and answer questions
 - Services: Variable depending on application, involves ongoing services to operate the application (e.g., updating data, maintaining models, etc.)
-

SAAS COST COMPARISON

SaaS does not just “push out” the costs to later years. In addition to reducing the upfront outlays, SaaS can dramatically reduce costs overall. The following example is based on a comparison of a P&C policy administration application.

	SaaS Model	Traditional Model
License Costs <i>The application directly plus operating system software, databases, other software necessary to support t application</i>	None	\$1,800k
Implementation <i>Initial data validation and loading, tailoring reports, training, feeds to and from risk management system, set up ongoing data update processes and controls</i>	\$500k over 4 months	\$1,500k over 12 months
Maintenance <i>Bug fixes, new releases, enhancements.</i>	None	20% of license fee, paid in advance, starting at license
Usage <i>Software use, software maintenance, software enhancements, monthly data updates, user help desk, hardware & infrastructure</i>	Month 1 – 4: \$0 Months 5 – 12: \$30k Months 13 – 24: \$40k Months 25 – 36: \$60k	None
Hosting <i>Hardware, communications, and other infrastructure to support applications</i>	None	Month 1 – 12: \$10 Months 13 – 24: \$12k Months 25 – 36: \$15k

Year 1 Outlay	\$740k	\$3,780k
Years 2 – 3 Outlay	\$1,220k	\$4,284k
Total First Three Years	\$1,940k	\$4,824k

For this scenario, the outlay in year 1 for the traditional approach is five times more than SaaS deployment, and over three years, traditional is over 2-1/2 times the cost of SaaS.