



How is Software-as-a-Service (SaaS) Different from Traditional Hosted Applications and Application Service Providers (ASPs)?

As part of the first wave of Internet-enabled applications, Application Service Providers (ASPs) became popular in the late 1990s. An ASP was a company that licensed a commercial software application (such as a document management system, ERP system, etc ...) or sometimes assumed a company's licenses for its own application and hosted that application in a secure, central facility. It then licensed that application to many companies and customized the package for each customer. The result was, theoretically, a customized version of commercial applications available to users at a lower cost of ownership.

But ASPs soon discovered that the cost of customizing and maintaining customized versions of commercial applications for each user was more expensive than originally predicted. ASPs, in many cases, did not control the feature set or development of the software they were deploying; their ability to make changes or innovate was limited. Customers, many of whom had moved mission-critical applications to the ASP model, soon discovered that ASPs lacked the domain knowledge to effectively customize the applications to the degree they expected, and that the resulting fees made it uneconomical. In sum, the cost and innovation advantages proved to be few. As a result of these limitations, the market for ASPs collapsed as part of the Internet bubble in 2001.

The Differences Between Traditional Hosted and SaaS Models

	TRADITIONAL HOSTED / APPLICATION SERVICE PROVIDER (ASP)	SOFTWARE-AS-A-SERVICE (SAAS)
Application Deployed	One-to-one: Each customer bears the entire cost of maintenance and customization.	One-to-many: The software, application integration and maintenance cost is dispersed among all users.
Customization	Burdensome: Traditional customization (often through programming) is rigid, expensive to change and increases delivery time (not to mention risk).	Rapid: A true SaaS application is configured, not customized. The result is a low-cost, low-maintenance application with behavior that adapts as your business evolves.
Implementation Timeframe	Long: Customization often means prolonged implementation. If additional products require integration at the hosting center, implementation is further delayed.	Almost Instantaneous: All customers use the same application. You can be up-and-running in record time.
Upgrade Frequency	Infrequently: Because the application is not multi-tenant, each customer's application must be updated individually.	Often: Several versions are released each year. All customers receive the upgraded application simultaneously.
Integration	Expensive and Extensive: Because ASPs customize each implementation for each customer, integration cost is borne by each customer. The cost to customize the application is high because many ASP applications require custom integration for each customer	Inexpensive: In SaaS models, functionality is integrated once. The integrated functionality is then made available to all users. Cost is amortized among all users.