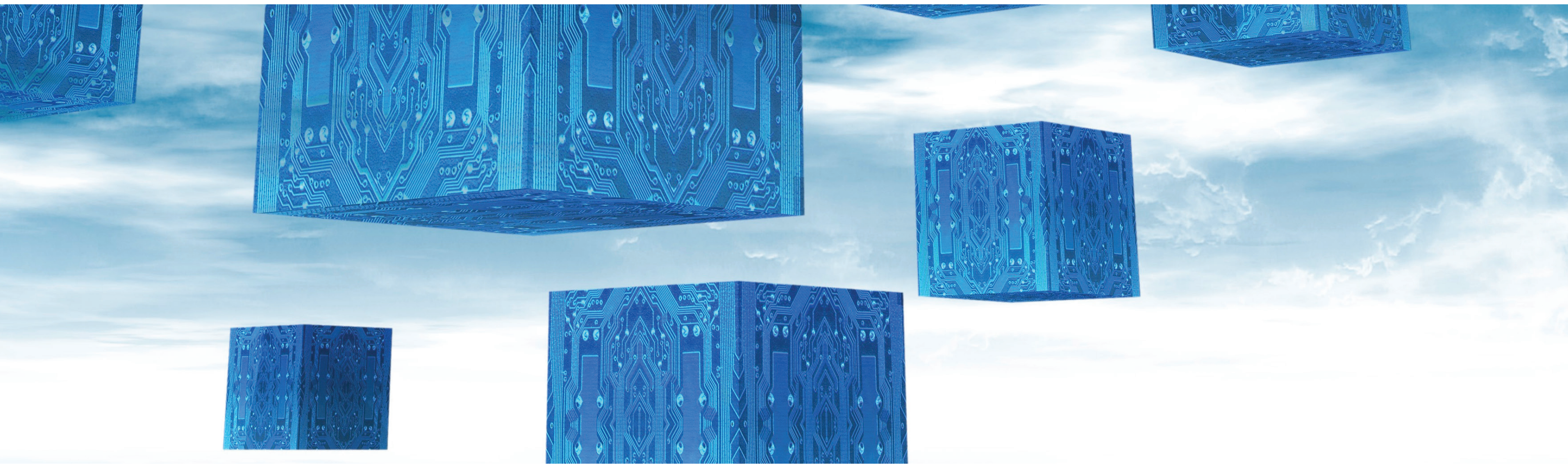
McKinsey&Company

JUNE 2015


© Victor Habbick Visions/Getty Images

Business Technology Office

From box to cloud

Six critical success factors can help software companies make the move from selling packaged products to offering online subscriptions and services.

Santiago Comella-Dorda, Chandra Gnanasambandam, and Bhavik Shah

Cloud computing is moving closer to the center of executives' strategy discussions. Concerns about security and decision rights remain, but faster processing speeds, better network connections, the ubiquity of mobile devices and big data, and other technological advances are making it more appealing for companies in all industries to purchase their software as a service (SaaS) rather than buy it in a box.[1] Under the SaaS model, companies access critical applications directly from central off-site servers; data capacity in the cloud is elastic, so companies can scale up or ramp down applications quickly, and they pay only for what they need. System upgrades happen automatically, so companies' applications become less costly and onerous to maintain.

As a result, some software providers are looking at adopting cloud-based delivery models for some or all of their offerings. According to International Data Corporation (IDC), the worldwide market for cloud-based SaaS offerings is expected to expand by about 20 percent per year through 2018, when it could exceed the $100 billion mark.[2]

Potential in the cloud

The software providers that have already implemented some form of SaaS delivery are reporting improved customer experiences, lower delivery and support costs, access to new markets, and opportunities to create new products. Adobe, for instance, is now delivering its publishing and design tools in a suite of cloud-based applications, dubbed the Creative Cloud. Customers pay a monthly fee to access applications such as Illustrator and Photoshop from Adobe Systems' central servers and to take advantage of online-only features, such as a social-networking site for creative professionals and content- and talent-search options.

But for many leading software developers, SaaS still remains something of an afterthought. Another recent report from IDC estimates that only 8 percent

Takeaways

Software as a service (SaaS) offers significant benefits, but customers and software providers have been cautious about moving to cloud-based computing.

Software providers may be nervous about the challenges of moving already-developed packaged software to the cloud—but companies that follow six strategic principles have a better chance of success.

The six principles focus on agility in the product-development process, management of customer relationships, the pace of product development, and companies' tolerance for failure and willingness to invest in new technologies and capabilities.

of the revenues of the top 100 software companies come from SaaS models, and seven of the ten biggest firms draw less than 5 percent of their software revenue from SaaS. Most likely, this is because a shift from boxed software to cloud-based applications and services entails significant challenges: Is the existing application architecture sufficient for supporting cloud-based delivery of products? Will the code base need to be refactored (a controlled technique for restructuring code incrementally) or entirely rebuilt? Does the software provider have the right product-development, operating, and distribution models in place for cloud-based delivery? Does it have the right people, with the requisite expertise in SaaS?

To answer these and other questions, CTOs, CIOs, heads of software development, and heads of application development would do well to consider several main technical questions and then emphasize six strategic principles for successfully switching to a cloud-based delivery model. The six principles focus on the process by which new products are developed and old products are reconfigured for the cloud, the ways companies should manage customer relationships, the pace at which product-development teams should operate, companies' tolerance of failure, and the degree to which companies are willing to invest in new technologies and capabilities. In our experience, companies that follow these principles stand a better chance of prevailing in a market in which the Internet is increasingly becoming a core mechanism for software delivery.

Six cloud-hopping principles

Many factors have contributed to SaaS's low adoption rates by software providers, but perhaps the biggest impediment is the design of existing applications: because packaged software typically was not developed with the cloud environment in mind, many reengineering questions must be answered. Our conversations with senior software development executives reveal a handful of critical technical concerns that companies must address before software and systems can change (see sidebar, "Cloud architecture: Picking the right approach").[3]

Cloud architecture: Picking the right approach

Software providers that want to switch to a cloud-based delivery model face several critical decisions relating to their code bases and application architectures. The optimal way forward will depend on their main objectives and starting point.

Go with a unified or separated code base?

Software companies need to determine whether it makes more sense to establish a unified code base for both their packaged and cloud software or to have a separate code base for each. This decision involves several considerations. Looking long term, for instance, is the company trying to build applications specifically for the cloud, or does it simply want to take advantage of certain aspects of the cloud environment—say, scalability, flexibility, and low cost—for development purposes? How long does it expect to continue offering both packaged and cloud-based software? And does its product-development team have the desire and attitude required to learn new technologies and unlearn previous coding practices?

A unified code base may be preferable if a company's current customers view the cloud as just another channel and if the company does not expect all of its customers to transition away from packaged software in the short to medium term, so the company has to maintain both versions of the software for the long haul. That was the case for a software company that decided to develop a cloud version of one of its products. The packaged version of its software would continue to have a significant customer base in the future, so to ease development and maintenance of the product, the team decided early on that it would use the same code base for both products and adopt a plug-in-based architecture for the cloud-specific components of the software. This decision allowed the team to utilize 90 percent of the existing code base for both versions of the software.

By contrast, a separated code base makes sense for organizations that see the cloud as a critical channel for future growth and expect to phase out packaged-software products sooner rather than later. In these instances, the company won't have to worry about building the exact same features in both cloud-based and packaged software products, because it will soon phase out the latter.

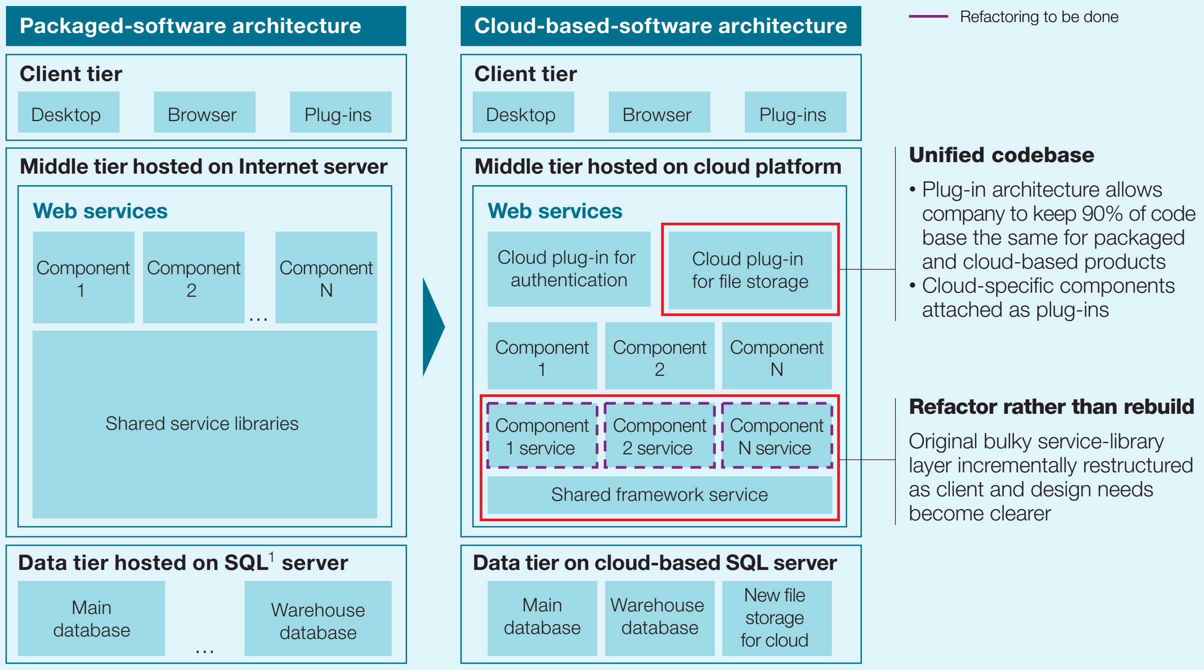
Develop a new architecture or refactor on the run?

Software providers that are transitioning to a cloud-based delivery model must decide whether to refactor their application architectures on the go (integrating incremental changes), as in the exhibit, or build entirely new ones from scratch. Critical factors in this decision are the pace at which the company is seeking to release cloud-based products and the capabilities of its existing application architecture. Refactoring is preferable if the current architecture is not quite ideal for cloud applications but still has some basic, required structural elements. A maker of payroll-processing software refactored some elements of its existing applications infrastructure so it could port certain applications to the cloud, but it left a few mature components as is, including some on the mainframe, to avoid incurring security or data-fidelity risks.

By contrast, establishing a new application architecture might make sense if the current code base is truly not transferrable to cloud applications. The organization might have a monolithic application architecture that demonstrates symptoms of "spaghetti" code—for instance, units of code that dictate business logic get mixed up with lines of code associated with the user interface. Or it may find that an architecture that was originally built to support packaged software cannot adequately scale up to the much larger number of users likely to retrieve cloud-based software online. Companies often build up "technical debt" from their prior architecture decisions, which is why a systematic assessment of current capabilities is so critical.

Exhibit

The move from box to cloud requires changes to application architecture.


Packaged-software architecture
Client tier
Desktop
Browser
Plug-ins
Middle tier hosted on Internet server
Web services
Component 1
Component 2
...
Component N
Shared service libraries
Data tier hosted on SQL[1] server
Main database
...
Warehouse database
Cloud-based-software architecture
Client tier
Desktop
Browser
Plug-ins
Middle tier hosted on cloud platform
Web services
Cloud plug-in for authentication
Cloud plug-in for file storage
Component 1
Component 2
Component N
Component 1 service
Component 2 service
Component N service
Shared framework service
Data tier on cloud-based SQL server
Main database
Warehouse database
New file storage for cloud
Refactoring to be done
Unified codebase
• Plug-in architecture allows company to keep 90% of code base the same for packaged and cloud-based products
• Cloud-specific components attached as plug-ins
Refactor rather than rebuild
Original bulky service-library layer incrementally restructured as client and design needs become clearer

[1] Structured Query Language.

These discussions also demonstrate the need for software providers to follow six core strategic principles.

Emphasize minimum viable products rather than 'big bang' releases

It can take up to two years for software-development teams to create new and "complete" versions of their packaged offerings for the cloud—software products or applications that are fully compliant with previous versions and that contain all the features customers want, plus additional ones designed to extend the usefulness of the software. Instead of following the traditional time line of releasing a "big bang" product every two years, companies should take advantage of the flexibility that a cloud-delivery model allows. They should plan on releasing minimum viable products (MVPs) to customers—"light" versions of software that are purposefully developed for testing and continual refinement. These MVPs can be released within months rather than years. Development teams can elicit useful feedback from customers about how these cloud-based offerings are working (or aren't), and they can adjust critical elements of the software accordingly. Companies can continually test their assumptions about the functionality of their products and determine whether they are delivering sought-after core customer experiences.

Treat users as part of the day-to-day development team

Building on the first principle, software-development teams need to engage with their customers early and often as they create new products for the cloud or reconfigure old ones. The cloud model allows companies to bring end users under the tent more easily and interact with customer segments in different ways. Software developers can run applications centrally for everyone, but they can turn features on or off for specific end users, gather their feedback, and adapt.

The cloud model also enables teams to roll out new software features in a controlled manner—testing them with, say, 2 percent of customers, then 5 percent if all goes well, and on until full rollout is achieved. Customers can provide feedback in real time—as soon as, or even before, applications or features go live. Software developers and product managers can ask customers to prioritize their needs via blogs when products are still in MVP phases, and they can codesign the full-featured version with them, engendering brand loyalty in many cases. As software-development teams collect more customer information, they can assess usage patterns and refine their A/B testing of certain software features and functions to determine which to include and which to drop.

Expect and tolerate failure

A cloud infrastructure brings many software-development benefits, including the ability to grow or shrink resources for an application in real time. However, the shared nature of cloud architectures can pose challenges because of factors that are beyond the software developer's control, such as hardware or network failures or network slowdowns. IT architectures and cloud-based applications must be designed to accommodate these potential outages, but even so, companies will need to expect and tolerate some failure. This is antithetical to the development of "boxed" software products, where risk-averse teams recognize how complicated and costly bug fixes and system patches can be to roll out, so they prolong their testing cycles.

By contrast, the cloud model enables easier maintenance, so developers should be given the liberty to release minimum viable products frequently and without hesitation, but with a mechanism in place for quick repairs. To deal with the potential for failure, some software providers have retooled their packaged products for the cloud to provide a "gracefully downgraded" customer experience—in the event of a network slowdown, for instance, removing all photos or other complex web-page elements—rather than a complete crash. Other companies have written automated scripts into their IT architectures that will, behind the scenes of daily operations, simulate random failures in the system and adjust parameters as needed. In this way, the company can test the systems' responses to failures that haven't happened yet but may eventually.

Adopt agile approaches to software development

Companies must acknowledge clear time-to-market differences between delivering software services in the cloud and shipping physical products. Given the expectations of accessibility and ubiquity associated with selling cloud-based offerings, software providers need to adopt a continuous-release mind-set—as we mentioned earlier, updating their applications frequently, rather than launching packaged products once every few years. This commitment to agile software development, in which requirements and solutions evolve through collaboration among self-organizing, cross-functional teams, can ease software providers' path to the cloud. Specifically, these companies will need to tightly integrate their R&D operations with their IT organization—an approach commonly known as DevOps, which brings both groups together to optimize product design, delivery, quality assurance, and maintenance.[4] Frequent, incremental releases can help the company reduce the complexity of software deployment, as well as the risk of large failures at the time of release. Companies can be more responsive to customers' needs and requests. A bug reported by one customer can be fixed quickly—so efficiently that other customers accessing the same application don't have a chance to see it or be affected by it. Our research shows that agile-software-development teams can increase their productivity by an average of 27 percent and boost the timeliness of feature releases by 30 percent (exhibit).

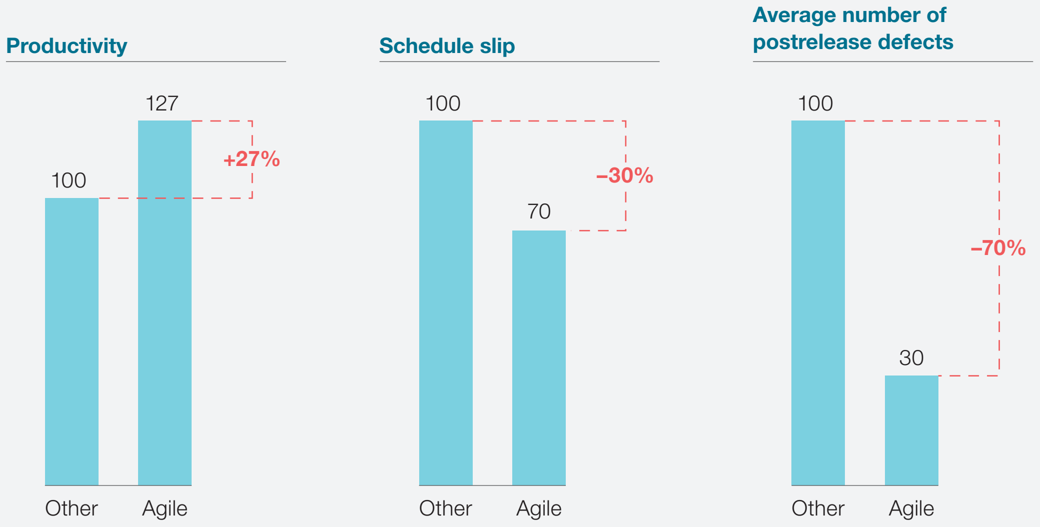
Give developers quality-assurance and testing responsibility

When the deployment of an update for a piece of cloud-based software fails, an immediate response is required. Companies cannot wait several months to issue the next release, as they can with packaged software; they must fix each problem as it occurs. The providers that have successfully transitioned their

Exhibit

An analysis of more than 1,500 projects suggests the value of agile.

Performance of teams using agile software development vs those using all other software-development methods,[1] % of 'other'



[1] Based on more than 1,500 projects in Numetrics industry software database.

Source: Numetrics; McKinsey analysis

packaged offerings to the cloud hold their software developers, not just their code testers, accountable for ensuring high-quality products and experiences. In this way, they can roll out critical fixes while simultaneously releasing new features and applications. This model is efficient as well: a developer can fix a bug introduced two weeks ago more quickly than a flaw introduced six months or two years back.

Because users are accessing applications in the cloud from multiple time zones, there is no optimal time to take servers offline to debug units of code, as companies might do with packaged software. So developers that are moving packaged products to the cloud should build advanced diagnostics and tracing capabilities into the software right from the start. Indeed, our conversations with developers that have successfully made the transition suggest that cloud-based products require three to four times more diagnostics capability than packaged software does.

Invest in cutting-edge capabilities

It should go without saying that making a successful switch to an SaaS model involves hiring top development talent who can inject new expertise into the organization at all operations and management levels—bringing in different perspectives on systems security, for instance, or capabilities in advanced analytics. Companies must also be willing to invest in the unique tools and infrastructure that will power a cloud-oriented product-development model. A DevOps model, for example, requires an advanced, fully automated testing environment in which developers can quickly try out code changes against various subsystems before releasing new features to customers.

■ ■ ■

Cloud-based SaaS currently remains a relatively small part of most leading software developers' product portfolios. But as customers' adoption of SaaS takes hold in the coming months and years, so should providers' attention to the box-to-cloud trend. The six principles described here can help developers successfully make this challenging shift. ■

[1] David Streitfeld and Nick Wingfield, "With Amazon atop the cloud, big tech rivals are giving chase," *New York Times*, April 23, 2015, nytimes.com.
[2] International Data Corporation's Worldwide Semiannual Public Cloud Services Tracker monitors vendors' business performance and reports on which companies are gaining or losing market share in various sectors and parts of the world.
[3] We examined recent software-company transformations and industry activity, and we conducted in-depth interviews with leading software companies that have transitioned products from traditional to cloud delivery.
[4] Satty Bhens, Ling Lau, and Shahar Markovitch, "Finding the speed to innovate," April 2015, mckinsey.com.

The authors wish to thank Alex Ince-Cushman, Akash Shah, and Tobias Strålin for their contributions to this article.

Santiago Comella-Dorda is a principal in McKinsey's Boston office; **Chandra Gnanasambandam** is a principal in the Silicon Valley office, where **Bhavik Shah** is a consultant.