


The New Mobile Architecture

A close-up photograph of a motorcycle's rear wheel and fender. The fender is bright orange, and the wheel is chrome with a multi-spoke design. The background is blurred with horizontal streaks, suggesting motion. The image is split horizontally, with the top half being a solid orange color and the bottom half showing the motorcycle details.

POWERING MOBILE APPS

Big changes on the client side and new capabilities on the cloud demand big changes in middleware application development and deployment.

The New Mobile Architecture

BY BILL APPLETON



Contents

SECTION 1

Three-Tier Application Deployment Architecture 3

SECTION 2

The Dawn of a New Mobile Architecture 6

ABOUT THE AUTHOR

Bill Appleton, CEO, DreamFactory 8

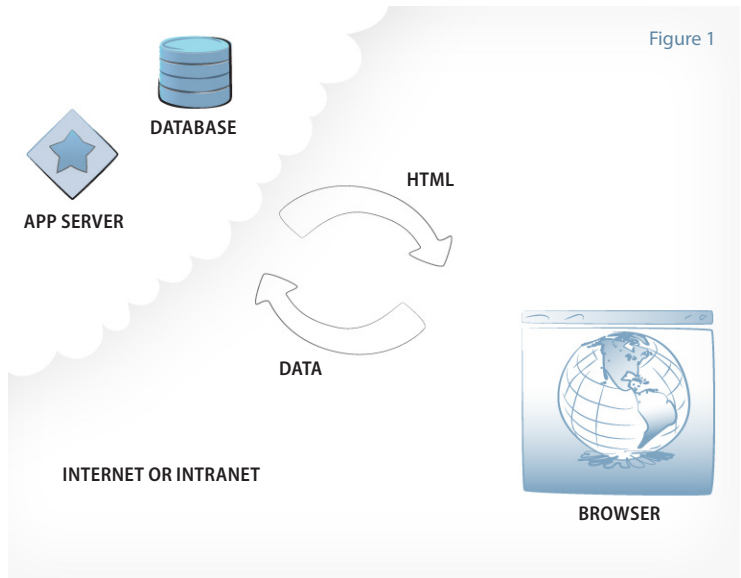
Quite a few enterprise software systems generate HTML pages on the server side. Examples include Websphere, Weblogic, JSP, PHP, Python, and Ruby. In many cases, these systems rely on a three-tier architecture consisting of a database, an application server, and a desktop web browser, all running on a persistent intranet or Internet connection. The business logic, application graphics, actual data, and page layout information are generated with server-side HTML pages and sent to the desktop browser. A typical application deployment scenario might look something like this:

Sophisticated server-side processing and storage capabilities are required.

A new HTML document—containing graphics, page layout, scripting and actual data—is generated for each screen update or user interaction.

A constantly available persistent connection with high bandwidth is required.

Typical Application Deployment Scenario



A few years ago, this picture started to change with the introduction of mobile devices and cloud computing. First, the desktop browser was sometimes replaced by a tablet or a phone. These devices have new capabilities—touch interface and geolocation information—but lack other features normally available in desktop browsers, such as screen real estate and access to the file system. Also, the mobile network is likely to be significantly slower and more expensive than a regular Internet connection. Mobile networks also exhibit sporadic connectivity as the user moves in and out of coverage or between zones.

Finally, the rise of cloud computing means that the server itself is no longer likely to run on-premises and might be virtualized in some way. Cloud

computing is a cost-effective strategy that allows the enterprise to scale for peak periods and pay only for the computing resources they actually use. For many companies, today's new application deployment picture might look something like this:

Application servers are expensive to run on cloud infrastructure.

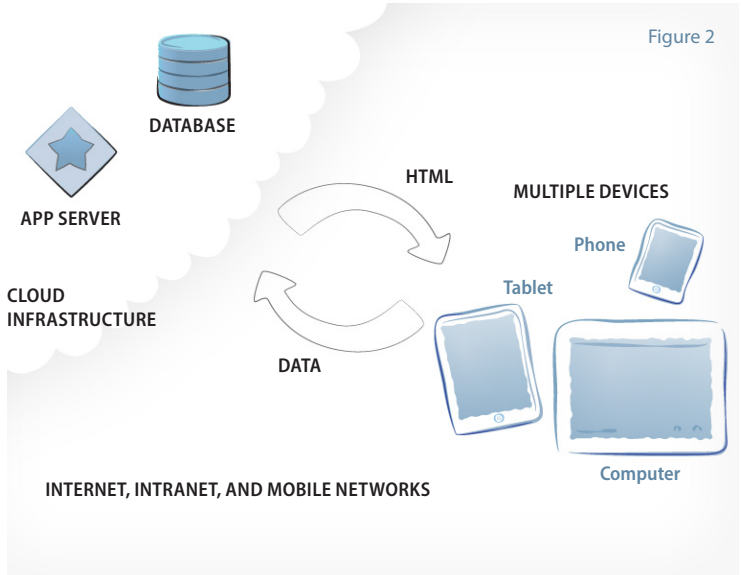
Legacy applications must be rewritten for mobile deployment.

Native consumer applications change expectations for enterprise apps.

Mobile networks are slow, expensive, and have intermittent connectivity.

Server-side HTML page generation results in poor user experience.

Today's Application Deployment Scenario



Some serious drawbacks are associated with this approach, however. First, since most of the heavy lifting is being done on the server side, the typical application server usually requires lots of system resources for processing and storage. This strategy can be expensive to implement on the “pay as you go” cloud infrastructure. Meanwhile, the application server is generating large HTML documents that contain graphics, page layout, and scripts, as well as actual data. These HTML documents produce a lot of content to download to a mobile device for each screen update, which can be expensive for users who don't have an unlimited data plan.

Intermittent connectivity is also an issue as mobile applications move between zones or into areas without coverage altogether. Since a new HTML page is often required for each screen update, the application dies when the connection is lost. This “click, get a page, click, get another page” model works well enough on a desktop browser with a persistent connection, but on a tablet or a phone, this model offers a cumbersome and heavy-weight solution that often results in a poor user experience.

The client-server pendulum has swung back and forth a few times over the course of the history of computing, and mobile devices are clearly a big reason that the action is shifting back to the client once again. Apple

pioneered “native” applications that run compiled outside of the browser. These applications are more likely to use JSON or XML data for collaboration than HTML pages.

Browser manufacturers also jumped on the “rich client” bandwagon with the development of HTML5, which has powerful features to enable client-resident applications that communicate through service transactions rather than HTML page generation. The HTML5 approach has the additional advantage of supporting any device. A variety of new JavaScript frameworks implement this development model for phone, tablet, and desktop. As the number of different mobile devices increases, we expect that the popularity of HTML5 will grow, especially for the enterprise.

This big change on the client side and these new capabilities on the cloud suggest the need for some corresponding big changes in middleware application development and deployment strategies. What we need is a comprehensive web services platform that can be installed on any cloud or data center and support a wide variety of mobile application development scenarios. This need is exactly the motivation for the DreamFactory Services Platform.

A service platform can conduct compressed document exchange transactions with a mobile client and achieve very large reductions in network bandwidth. Since the services are atomistic and under client control, occasionally connected applications are also possible. And because the interfaces are based on popular web standards, many different client environments are supported, including HTML5, native application, and server-to-server communication. With a service platform, we can achieve a much more rational system for developing and deploying mobile applications.

Service platforms leverage cloud infrastructure in a cost-effective manner.

Mobile browser applications can use services through HTML5 with REST and JSON.

Native consumer applications can use collaborative cloud services with XML or JSON.

Overall network bandwidth is dramatically decreased, lowering deployment cost.

Occasionally connected application strategies become easier to implement.

Service-based applications dramatically improve the user experience over mobile networks.

New Application Deployment Architecture

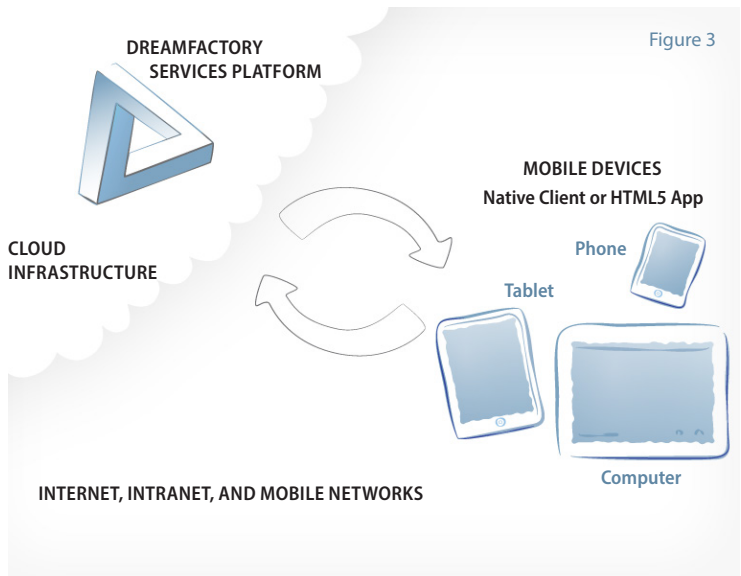


Figure 3

In the old model, all the graphics, business logic, and user interface must be generated on the server. But when we move to a rich-client model, we have an opportunity to replace that infrastructure with a prebuilt and standards-based service platform. Based on our experience building rich-client

applications, we know that the right palette of services will support a very large number of application scenarios. Since the services interface talks directly to the database, the speed and performance of the system is maximized. This increased performance dramatically reduces the cost of expensive mobile data plans. And because this model doesn't require user experience or graphics generation on the server, it is much less expensive to deploy on a pay-per-use cloud server. The next diagram shows the service palette available on the DreamFactory Services Platform.

User administration enabled for single sign-on, password hashing, user roles, and permissions.

BLOB data interface hides credentials on the server and provides binary document management.

Application hosting enables movement between platforms and reduces cross-domain issues.

External services provide integration with existing enterprise data systems and new services.

Open source model enhances security and enables installation on any cloud or data center.

DreamFactory Services Platform (DSP)

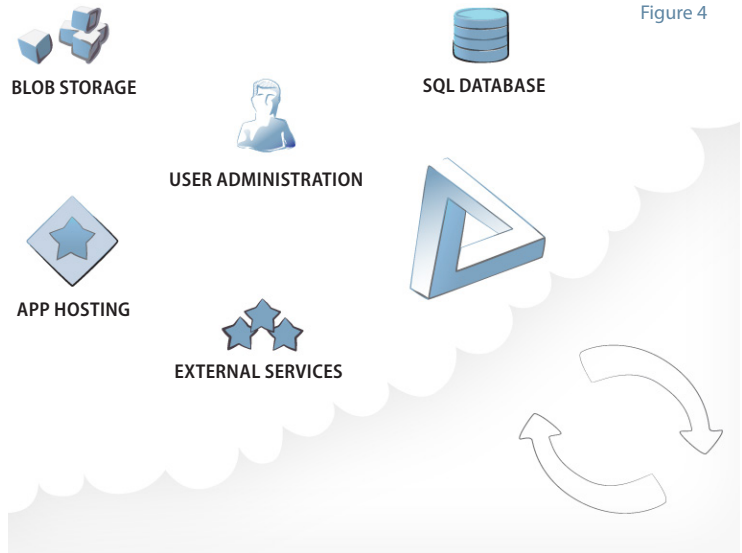


Figure 4

One of the most exciting aspects of the new mobile architecture is that once business logic and application graphics are moved to the client, the server-side software environment becomes much less complex. Essentially the service platform becomes a standards-based way to expose all of the various cloud assets required for application development. This architecture opens the way for the entire server-side stack to become a standardized commodity capable of supporting a very wide variety of mobile application development scenarios.

The new mobile architecture enables a small team of front-end developers—or even a single engineer—to build a state-of-the-art mobile application without the need to create any server-side software. The traditional API haggling between the client and server teams also disappears. This application development model promises large savings in time to value, software development costs, cloud hosting fees, and mobile network bills, while at the same time dramatically improving the user experience for mobile applications.



CONTACT BILL

[@bill_appleton](#)

[linkedin.com/in/billappleton](#)

[billappleton@dreamfactory.com](#)

[Learn About DreamFactory](#)

[dreamfactory.com](#)

About Bill Appleton

Bill Appleton is a leading expert on the client-side use of cloud services and the development of rich-media authoring tools. He has designed and written approximately three dozen professional software publications, including the first rich-media authoring tool, *World Builder*; the groundbreaking multimedia programming language *SuperCard*; the number one best-selling CD-ROM *Titanic: Adventure Out of Time*; the *DreamFactory Player* for consuming cloud services; the first AppExchange application *DreamTeam*; and the first third-party enterprise applications for Intuit *WorkPlace*, Cisco *Webex Connect*, and Microsoft *Windows Azure*.