DeNovo FinTech Focus

Banks can regain tech advantage through serverless computing

PwC's DeNovo: A platform that helps you understand how disruption affects business strategy and what actions to take

Banks—well known for their monolithic information technology environments—may regain their technology advantage by embracing the smallest computing form yet: serverless computing.

pwc

Software moves from build-it-and-they-will-come to just-in-time models

Every bank today is first and foremost a software company. The quality of the consumer experience, the simplicity of interaction, the effectiveness of trading processes, and the efficiency of the back office all depend on the design and deployment of software and play an integral role in competitive differentiation.

This is not good news for most established banks. They face a significant structural disadvantage against their high-tech competitors, especially in developing applications that fit with their legacy information technology (IT) infrastructure. The limits of legacy systems are restraining the financial services industry from innovating at the required speed to keep pace with changing consumer expectations and FinTech startups.

Although there is no simple solution, serverless computing may be the way forward for financial services firms. With this technology approach, software applications are stripped into individual functional "events." These events are portable and cost-efficient. Most important, they are not tethered to legacy infrastructure environments. In other words, they are "serverless." By building applications without the operational infrastructure burden, banks can untether their software development from legacy infrastructure and reclaim their technology advantage. This form of computing is the best opportunity yet to close the development gap with FinTechs.

Software development is moving from a build-it-and-they-will-come model to one that is "just in time" because the costs and basic units of computing continue to shrink. Virtual machine–based infrastructure now is moving to software containers and soon will entail only individual tasks or functions.

Senior banking executives need to come to terms with this change, its challenges, and (especially) its opportunities.

Serverless computing provides an opportunity for organizations to relinquish legacy infrastructure and take full advantage of the public cloud, rather than just adopting software-as-a-service applications. If your organization has not yet started to rethink its software development framework and processes, you are late to the game and will fall further behind FinTech innovators.

Serverless provides an opportunity to relinquish legacy infrastructure and take full advantage of the public cloud

Time for banks to disrupt banking IT

The rapid speed at which FinTech startups introduce new services is the primary reason financial services firms are vulnerable. Though the banking industry has taken initial steps to respond, the pace of application development by banks remains slow and hinders innovation. Financial services firms of all sizes need to adjust. There are two ways in which incumbents can reclaim their technology advantage:

1. **Capitalize on your industry expertise.** Banks own more consumer data than startups do. They need to learn to mine this trove of information and align it with their software development efforts to produce differentiated services and customer experiences.
2. **Accelerate the pace of software development.** Software is at the center of all innovation, and FinTech startups' rapid pace of product development is a threat to the traditional practices of the financial services industry. To better compete, incumbents need to redefine processes, shed IT operations and infrastructure, and realign greater investment to application development. In other words, incumbents need to disrupt their own IT organizations.

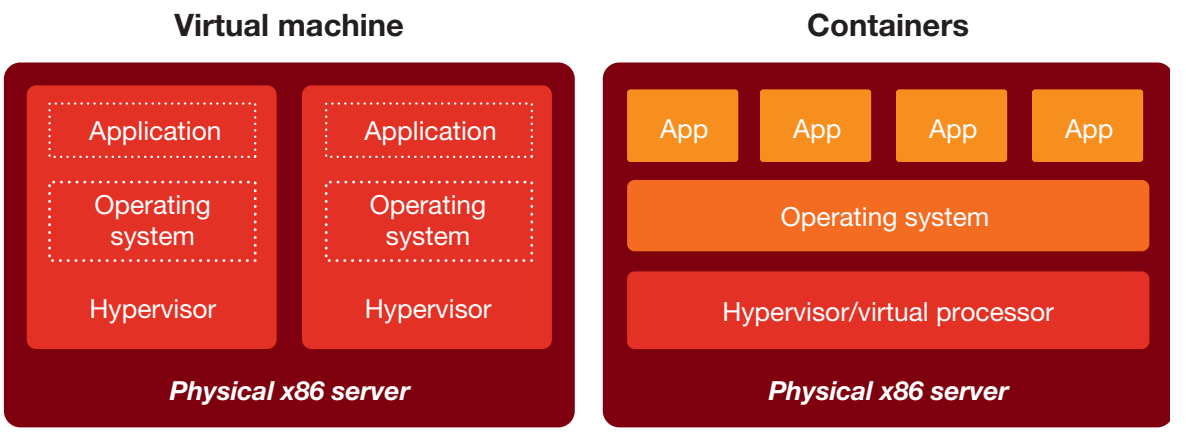
Containers are small, efficient, and popular

A newer computing form—software container technology—has seen significant adoption across mainstream organizations over the last two years. Containerization involves splitting applications into small parts, or containers, all sharing the same underlying operating system.

Unlike the virtual machines that preceded them, containers do not require individual operating system instances. Thus, they are smaller and portable, which introduces further infrastructure efficiency (see Exhibit 1). Many organizations have seen a 5:1 to 15:1 reduction in server requirements through the use of virtual machines. Containers take this a step further by sharing the virtual processor and operating system to expand application density per server unit upward of 100:1, or a 10-times increase relative to a virtual machine approach.

Exhibit 1

Application density per server



Source: DeNovo

Application portability is a key feature of software containers

Application portability is a significant aspect of software containers. Once the application is containerized, its small form enables it to move from an organization's data center to various computer environments, creating a viable way to migrate applications from legacy infrastructure to the cloud. A container-based approach also follows a standard format from internal data centers to public cloud environments. The format was developed in part by the Open Container Initiative, an open governance organization formed under the Linux Foundation that defines how the software will operate reliably across a range of computing environments.

Efficiency, lower cost, and portability have made containerization massively popular. Docker, the leading provider of software container solutions, has seen more than

2 billion downloads of its product since its launch in March 2013.[1] Organizations are using containers to reassemble legacy system applications so as to migrate them to a cloud environment; any application developed in the last 15 years can be moved into a container.

Many large internet companies have widely adopted containers

The efficiency of containers is one reason that cloud infrastructure companies such as Amazon, Google, and Microsoft have championed their use. Google, for example, operates entirely in containers, starting more than 2 billion containers per week. This provides a consistent underlying infrastructure, allowing the company to maintain a limited number of operating system versions across its entire computing environment. The setup translates directly to a vastly lower support cost, given the small staff required to maintain the operating system environment.

The development of the software containers is in large part happening through new entrepreneurial software companies that have often spun off from major Internet companies (see Exhibit 2 on next page). Docker and Kubernetes, for example, are two of the most widely used container solutions, and both descend directly from Google. Perhaps sensing the importance of container use by mainstream industries, Google has expanded support of container systems beyond its own proprietary system (lmctfy) to solutions such as CoreOS rkt. This support of third-party solutions could spur broader competition and ultimately drive greater innovation.

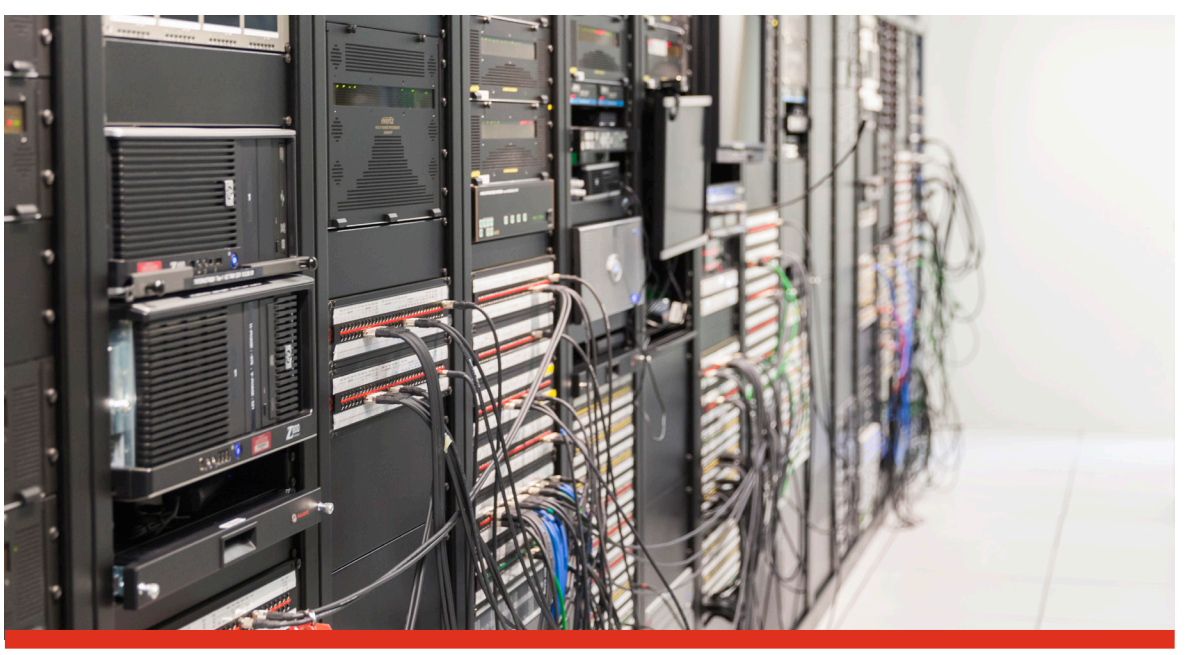


Exhibit 2

Container solutions and their Internet "parents"

Container management systems

AWS EC2 Container Service: A proprietary container management system for Amazon Web Services instances.

Azure Container Service (ACS): An overlay management system that manages third-party container systems including Docker Swarm, Kubernetes, and Mesosphere. Microsoft open-sourced the core engine behind ACS.

Docker Swarm: Docker's container management system that was a natural expansion given the share of Docker containers.

Kubernetes: Open source system based on Google's internal Borg system.

Mesosphere: Open source management system that is backed by more than 60 companies. Recent addition of Kubernetes support positions the platform as a neutral player.

Container engines

CoreOS rkt: A relatively open (rkt containers can be ported to other container systems) and security-focused container engine.

Docker: The leading containerization platform initially developed in 2008 and released as an open source project in 2013.

Google lmctfy: Open source version of Google's container platform, which has subsequently been deemphasized by the company.

LXC (Linux Containers): Containers native to every Linux distribution.

Source: DeNovo

Serverless comes with an entirely different economic model

Containers and serverless computing

The concept of containerization is now being extended to serverless computing, which offers the greatest separation yet from the supporting infrastructure stack and potentially a clean break from legacy infrastructure. This new technology approach, however, comes with an entirely different economic model. It starts with even further fragmentation of the software stack into small components and the advent of events: The economic model is entirely different, which will influence adoption.

Separating the infrastructure from the application logic will significantly change the staffing and economic models by increasing the effectiveness of smaller developer teams. A serverless approach separates the infrastructure functions from the software development process, allowing teams to focus exclusively on new app development and not on operations. This enables a small team to have an outsized impact on new app development.

Effective small development teams bring a new economic model

If incumbents learn only one thing from FinTech (or tech in general), it's how small development teams can deliver massive value. Stripe, for example, has fewer than 600 total employees[2] and is valued at $9 billion.[3] Facebook acquired WhatsApp for $17 billion[4] (55 employees), and VMware acquired Nicira and its 70 software developers for $1.3 billion.[5] JPMorgan Chase, in contrast, has 18,000 software developers, yet two-thirds of its $9 billion technology budget is tied to support and maintenance.[6]

Serverless computing can also convert IT cost solely to application use. Organizations that deploy serverless applications on cloud environments pay only when the code is utilized. Serverless applications contain the bare minimum requirements to operate and are active only when triggered by consumer use, in which case the full infrastructure stack (server, storage, networking, etc.) is launched to execute the task and then collapsed as the task is completed. There is no up-front cost for infrastructure and no fee when the code is idle. This model allows IT costs to be tied just to the use of the application.

Serverless computing will thus enable banks to deliver consumer services at a more reasonable cost. This model will facilitate the outsourcing of all IT operations, maintenance, and support to a cloud provider, which, in many cases, will free up 50 to 60 percent of the total IT budget. It is the clearest path to date for organizations to make the "all in" migration to the cloud.

A startup-like economic model

Serverless computing will give incumbents access to the same economic model as startups: a full, global-scale IT infrastructure with no up-front costs. This will level the playing field, but only if incumbents acknowledge the disadvantage of current infrastructure and move to change.

The small units of microfunctions in a serverless model are completely counter to the monolithic application architecture that many banks have in place. The economic model with these smaller computing units more closely resembles a true utility model; expense is incurred only when the application is in use, and fees are not incurred for redundant servers, virtual machines, or storage (see Exhibit 3).

This new serverless model delivers on the promise of utility computing but in a much more modular and lower-cost fashion. It introduces massive efficiency into the IT model (including as much as 10 times the efficiency of a virtual machine approach).

Exhibit 3

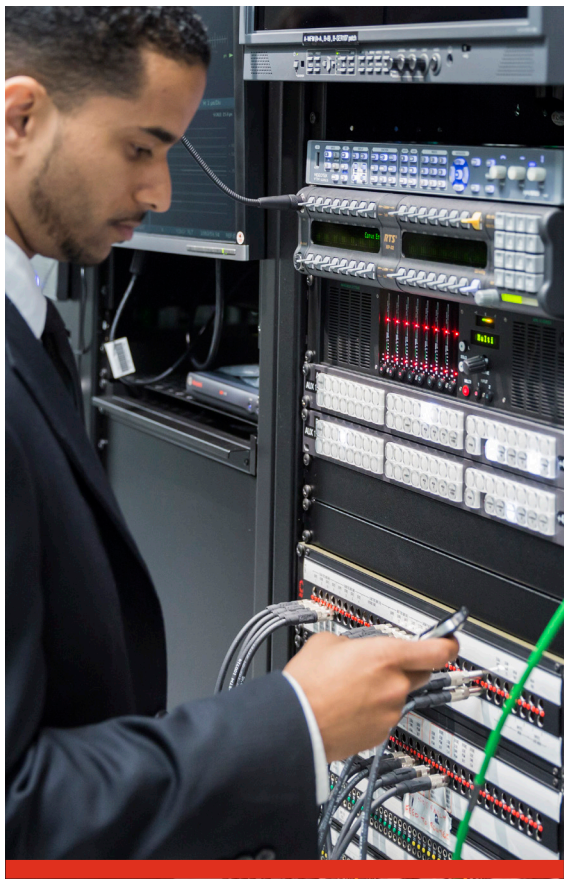
"Lights on" pricing models

Product	Details
AWS Lambda	Pricing is based only on the compute time Lambda consumes. Lambda compute time is based on when the function starts to execute in response to a trigger, and there is no charge when the code is not running.[1]
Google Cloud Functions	Pricing information for Cloud Functions is not available. The product is currently in alpha.
Microsoft Azure Functions	The Azure Functions consumption plan is billed based on resource consumption and executions. Microsoft does not impose up-front costs or termination fees and bills only for compute time used when the event is running.[2]

1. https://aws.amazon.com/lambda/ pricing/
2. https://azure.microsoft.com/en-us/pricing/details/ functions/

Source: DeNovo

In addition to increased flexibility for innovation, a serverless model can redefine mundane tasks. Banks run thousands of reports—daily mark-to-market, weekly, monthly, quarterly, and other documents. Report production typically requires development staff, DevOps, dedicated infrastructure, disaster recovery planning, and compliance because of the sensitive nature of the information. With a serverless approach, the reporting can be considered a function-as-a-service, possibly running only for a few hours every week. Once completed, the reporting function is terminated with no possible access to the data—powerful from both economic and compliance perspectives.



The path to serverlessness

To be sure, financial services firms face significant challenges in competing against technology-centric companies in this new IT realm. And they are right to be thoughtful. Many banks have considered cloud environments and container-based software but have chosen to remain with, or are reluctantly tethered to, on-premises legacy infrastructure. This gives them a bipolar model—expanding to a public cloud in a piecemeal fashion while also remaining on traditional infrastructure.

Serverless computing will transform the deployment model to the specific function and fully align the economic cost of technology with consumer use. And this could ultimately accelerate what has been a slow but continual migration to a public cloud environment.

Serverless computing: A legacy infrastructure exit strategy

Over the next several years, serverless computing may define the next chapter in IT (see Exhibit 4). While containers are revolutionary in how they deploy applications, a serverless approach is more concerned with deployment of individual event functions. Although it is true that both container and serverless computing enable continual migration to the cloud by fully detaching the underlying infrastructure from the application, a serverless model accelerates this migration and redefines the economic model because it focuses on individual functions and not on applications.

Exhibit 4

Severless event-driven platforms

AWS Lambda: An early serverless solution (previewed in November 2014), it has helped define the serverless concept.

Azure Functions: The release of Azure Functions in March 2016 marked Microsoft's entrance into a serverless architecture.

Galactic Fog: The company developed a quasi-abstraction layer on top of the traditional container environment in an effort to move containers to the cloud in a serverless fashion. In addition to its own event-driven engine, Galactic Fog has focused on scalability for the public cloud.

Google Cloud Functions: Google's entrance into serverless platforms came in February 2016; Cloud Functions is in alpha.

IBM OpenWhisk: An open source variant of a serverless event-driven engine released in February 2016.

Iron.io: A serverless platform framework that aims to integrate various third-party solutions (Lambda, Docker) to reduce vendor lock-in.

NStack (formerly StackHub): A serverless platform that differentiates with its focus on developer-friendly tools.

Serverless: Initially focused on AWS Lambda applications but has expanded support to other solutions.

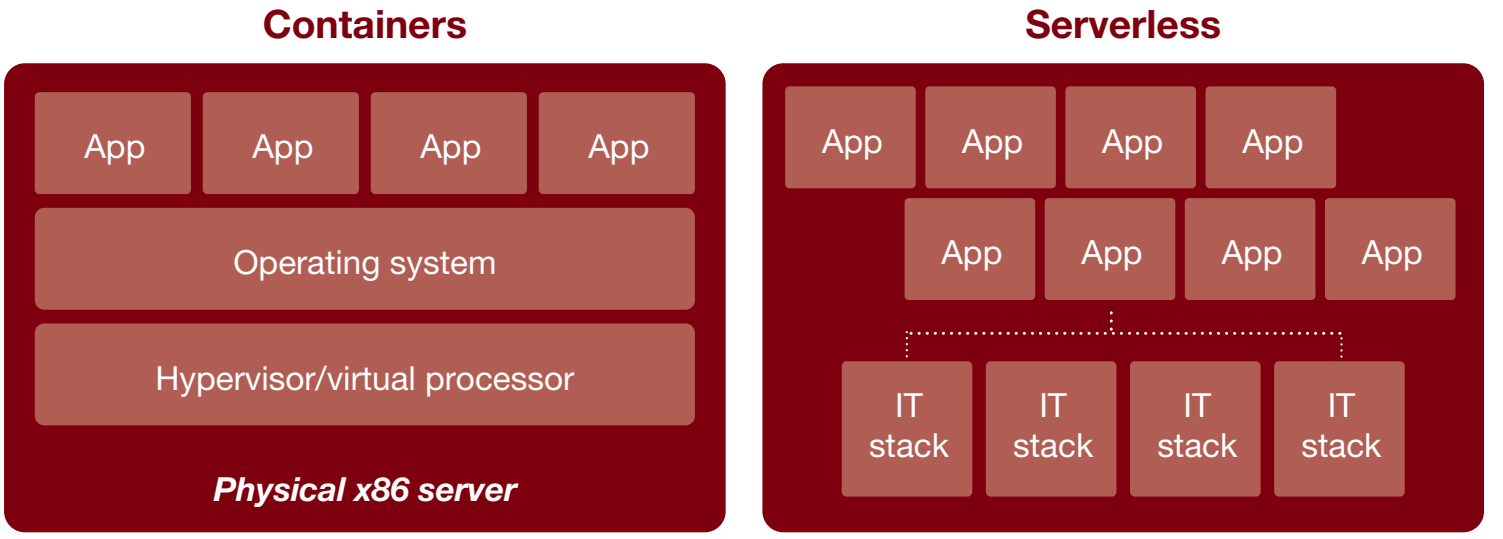
Source: DeNovo

A serverless model reduces the application to individual event tasks or microfunctions (see Exhibit 5). Individual microfunctions run on any available infrastructure in the computing and public cloud environment, making computing resources essentially unlimited. Containers, in contrast, run on the infrastructure available to them.

In short, the serverless model of software development focuses on the functional output and not on how it will run. The separation between supporting requirements (e.g., server configuration, storage, networking, scaling concerns) and the software development process makes serverless computing different from IT functions of the past. Here lies the greatest opportunity for organizations to modernize their applications for a public cloud environment or to unshackle application development from the restraints of legacy infrastructure.

Exhibit 5

Containers vs. serverless computing



Source: DeNovo

Endnotes

[1] https://blog.docker.com/2016/02/docker-hub-two-billion-pulls/

[2] https://stripe.com/press

[3] www.wsj.com/articles/stripes-valuation-nearly-doubles-to-9-2-billion-1480075201

[4] www.sec.gov/Archives/edgar/data/1326801/000132680115000006/fb-12312014x10k.htm

[5] www.wired.com/2012/08/why-vmware-paid-1-26b-for-70-software-engineers/

[6] www.jpmorganchase.com/corporate/investor-relations/document/firm_overview_investor_day_2016.pdf

Contact us

For additional information about PwC's FinTech practice or the DeNovo platform, please contact:

Haskell Garfinkel
US FinTech Co-Leader
+1-408-534-4727
haskell.garfinkel@pwc.com

Dean Nicolacakis
US FinTech Co-Leader
+1-330-283-4986
dean.nicolacakis@pwc.com

Aaron Schwartz, CFA
Head of Research, DeNovo
+1-646-647-4060
aaron.m.schwartz@pwc.com

Global team

Yair Weisblum
Partner, PwC Canada
+1-416-814-5892
yair.weisblum@pwc.com

Henri Arslanian
Director, PwC Hong Kong
+1-852-2289-2490
henri.arslanian
@hk.pwc.com

Steve Davies
Partner, PwC UK
+44-0-131-260-4129
steve.t.davies
@uk.pwc.com

Rei Tanaka
Partner, PwC Japan
+81-90-7280-2652
rei.r.tanaka@jp.pwc.com

John Shipman
Partner, PwC Australia
+461-2-8266-0198
john.shipman@pwc.com

Andrew Taggart
Partner, PwC Singapore
+65-6236-4548
andrew.taggart
@sg.pwc.com

Jianping Wang
Partner, PwC China
+86-21-2323-5682
jianping.j.wang
@cn.pwc.com

Banking services

George Hodges
Subject matter expert
+1-678-449-6278
george.hodges@pwc.com

Musarrat Qureshi
Subject matter expert
+1-917-674-1609
musarrat.qureshi@pwc.com

Thomas LeTrent
Research analyst
+1-202-729-1672
thomas.letrent@pwc.com

Capital markets and investment and wealth management

Michael Raneri
Subject matter expert
+1-415-519-4906
michael.raneri@pwc.com

Insurance

Jamie Yoder
Insurance advisory leader
+1-773-255-2138
jamie.yoder@pwc.com

Christopher Martin
Research analyst
+1-646-471-2122
christopher.martin
@pwc.com

Transaction and payment services

George Hodges
Subject matter expert
+1-678-449-6278
george.hodges@pwc.com

Michael Landau
Research analyst
+1-646-471-0878
michael.landau@pwc.com

Blockchain and smart contracts

Subhankar Sinha
Subject matter expert
+1-646-379-3721
subhankar.sinha@pwc.com

Joseph Yoo
Research analyst
+1-916-764-2855
joseph.h.yoo@pwc.com

www.pwc.com/denovo