xignite

White Paper

The Market Data Microservice Revolution

It's been more than 17 years since Xignite launched its first API and more than 10 since we announced our first AWS-based solution, and still, the world of market data remains riddled with legacy technology: firehose feeds and FTP files are still the norm for data distribution. But this is about to change. The logistical challenges created by the pandemic will likely give larger financial institutions the impetus they need to accelerate their transition to the Cloud. And nothing is likely to have as much impact on the market data industry going forward as the deployment of microservices.

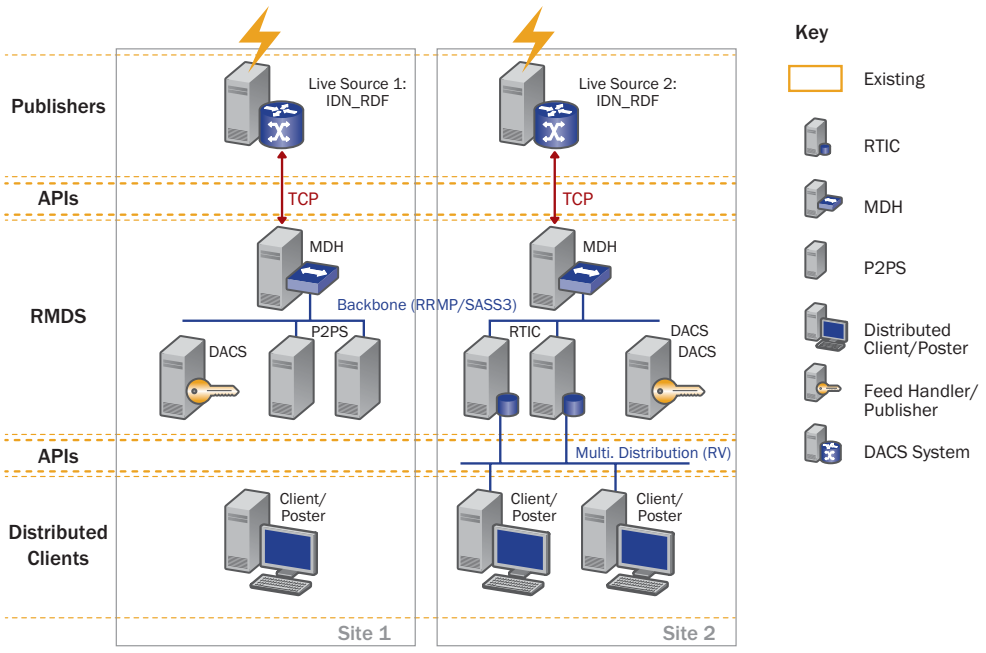
Flashback to Legacy

Most of the technology used by financial institutions has its roots in the pre-internet early 90s (think Triarch/Tib) and although this technology was upgraded over time, the core architectural principles behind it never really changed: a multi-tiered, on-premise tech stack designed to carry data from high-bandwidth dedicated private networks via proprietary protocols down multiple layers of on-premise distribution technologies (collocation, distribution servers, feed handlers, access points, data hubs, message buses, etc.). The net results (after years of iterative layering of new functionality) has been a complex hodge-podge of proprietary software and hardware that has limited compatibility from version to version, requires extensive experience and training to maintain, is overkill for basic data distribution needs, and simply cannot easily be upgraded or unplugged without putting the business at risk.

Total Lack of Innovation

To be fair, the reasons behind such heavy architectures were legitimate. At that time, there was no bandwidth capable of handling the market data volumes of the time, there was no cloud or grid computing, there were no widely accepted interoperation protocol, and there were only very basic Application Programming Interface (API) standards. In other words, technology was just not far along enough to enable alternatives.

But those alternatives have been around for more than 10 years now, and legacy vendors have failed to make any reasonable progress towards upgrading their technology for the cloud. They have been held up by backward compatibility requirements, concerns about revenue preservation, lack of technology expertise and their own internal issues and battles.



Typical Multi-Level Legacy Market Data Infrastructure

(Source: Wikimedia Commons)

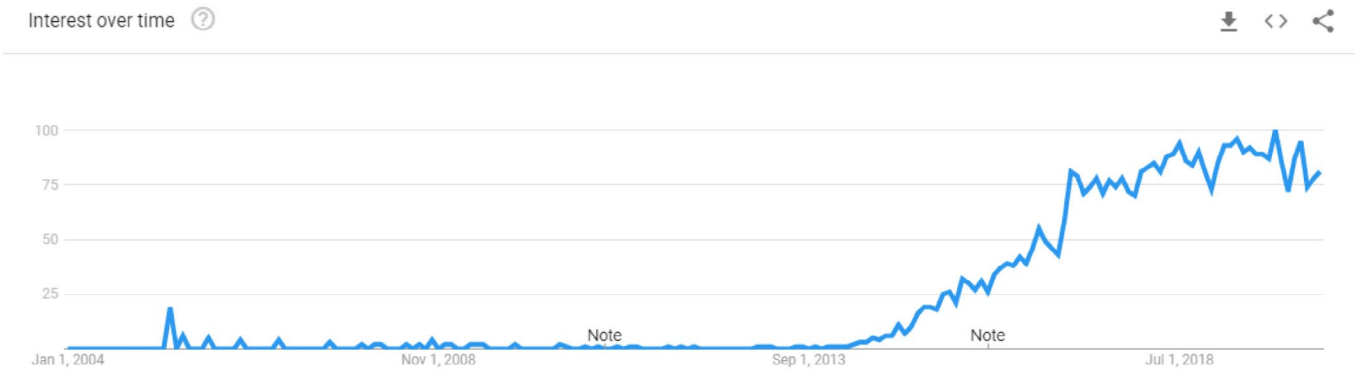
Emergence of Microservices

Fast forward to a different era. Now, internet bandwidth is reliable and cheap, RESTful APIs have become the universal way to integrate systems, and cloud computing scalability, reliability and functional depth has matured beyond the skills and capabilities of anything any firm can do on their own.

The most critical of those evolutions is the emergence of microservices. Given the nature of market data distribution, microservices have the potential to turn our industry on its head.

Microservices are not new. According to Garner Group, they reached the peak of inflated expectations in 2016. And if you look at Google Trends for the term, it's clearly been a steady subject of interest for several years. This means that they have had time to mature and become able to challenge the status quo.

Google Trends—Query for "microservices" since 2004



Microservices Defined

So what are microservices? Initially devised to solve the inherent problem of monolithic design, microservices break up applications into a suite of smaller, loosely coupled and independently deployable and scalable services each running in its own process and communicating with lightweight mechanisms, often an HTTP RESTful API. These suites of small services are each designed around different and specific business capabilities.

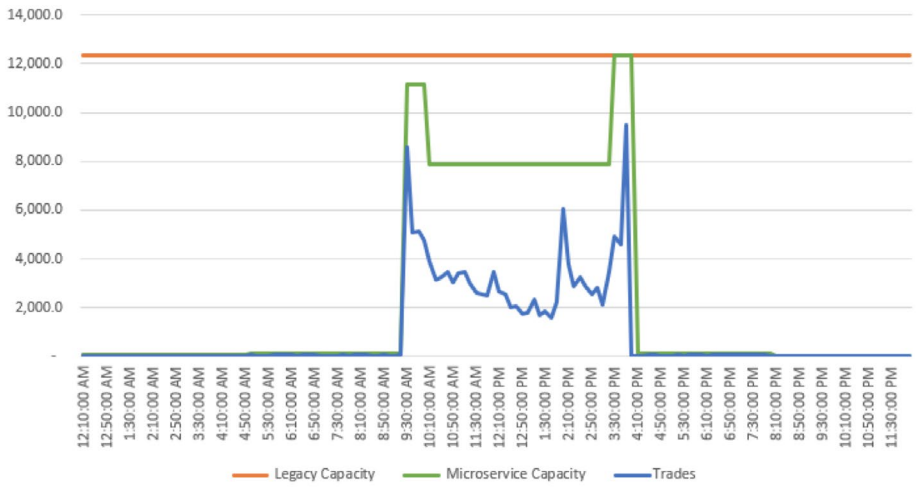
In contrast, traditional market data infrastructures were designed as monoliths. They ended up bloated, complex and expensive to operate. They did not support new integration standards but instead promoted obscure proprietary standards and ended up creating vendor dependency and spawning an inextricable web of point-to-point interfaces. Another problem with monolithic architectures is that deploying anything requires redeploying everything. So even small tweaks to tangential functionality carry a major tech and business risk. The inability to change these systems quickly is what has frozen the ability of firms to innovate, and this significantly enabled emerging fintech firms (who were not shy about adopting REST APIs and cloud-native technology) to out-innovate larger financial institutions in the last decade.

The Game Changer: Low-Cost Scalability

The primary reason why microservices are going to be so impactful on the world of market data is the low-cost scalability they afford. Capital markets tend to be peaks and lows operations. Even for global firms operating around the clock, the market open and market close times in New York, London or Tokyo generate huge peaks in inbound messages from the markets and demand for data from users and systems (terminals, trading systems, investors, etc). With a monolithic architecture, there is zero elasticity to the infrastructure so you must provision it for the peaks. If not, delays occur—and market data delays can cost millions of dollars. Market data volumes during peak activity can be massive. Provisioning and maintaining an on-premise infrastructure for peak activities therefore is very expensive and it is not uncommon for the peaks to push the limits of what that legacy technology was designed for. In other words, legacy market data technology is not very horizontally scalable. It cannot be easily distributed and it can only scale slowly.

In contrast, microservices are designed for infinite horizontal scalability and quick elasticity. You can more effectively scale for growth and peak in users and transaction volumes, since the infrastructure can be easily horizontally replicated. This means that if you design your services properly (for example using container technology allowing for easy horizontal scaling coupled with appropriate orchestration management tools), you can bring up the capacity you need when you need it, and you can shut it down when you don't. As you result, you only pay for the capacity you use, not the one you might need. If a pandemic is bringing huge volumes to the market, you just crank that capacity a bit more. Additionally, if your microservice is granular enough, you can truly fine-tune the amount of capacity you bring online, further lowering your costs.

The chart below shows the value of "just-in-time" capacity management capabilities in the Cloud. Demand for market data (i.e. capacity to server market data) is heavily correlated with market data volumes. So for instance, the requests for quotes originating from all parts of the firm are much higher right after market open and right before market close. The blue line on the chart shows typical volumes of trades for US equity markets. The orange line is the capacity you need to provision with a legacy infrastructure. Because this is non-elastic, on-premise capacity, it is fixed and it must have some cushion to deal with market bursts (30% here). The green line is the capacity you can provision using microservices. If you know the patterns of the market, and if your microservices are well designed so that they can auto-scale quickly, you can provision close to the market need. Over a 24-hour period, the green line represents only 20% of the capacity required without microservices. Even if you can achieve economies of scale with on-premise hardware vs cloud, the advantage of microservices is clear-cut.



To be fair, achieving just-in-time capacity management with microservices requires advanced knowledge and experience with cloud technology and it is not something most firms are ready for, but it is critical to proper microservice cost management in the Cloud.

Functional Granularity and Simplicity

The second reason why microservices are so key to the world of market data is the fact that they enable firms to deploy only the functionality they need. Because microservices are loosely coupled and can operate independently of each other, they do not require a significant functional footprint forcing the firm into a big-time migration. With a monolithic market data platform, it's an all or nothing deployment. You cannot have just one component and not the others. You will need some servers for co-location with feed handlers. You will need some distribution servers. You will need some access points servers and some message buses servers. You get the complexity and the big footprint from the start. You cannot easily carve out the functional complexity you don't need.

With microservices, you can start slow and migrate easily. Each microservice is single-purpose and runs a unique process to serve a business goal. Let's say you want to get started with limited market coverage and limited consumption and that you don't need some features like reference data management and entitlements because your use case is very controlled. You could easily only deploy the services you need. There will be no hardware deployment involved. There will be very limited functional complexity. Integration will be quick due to the use of open API standards. You can mix and match technologies and vendors since the interfacing is done using open standards. Deployment will be drastically simpler than implementing a traditional market data platform. Microservices even offer greater coding language freedom and make apps easier to refactor or replace going forward.

It is actually fascinating to see how some large financial institutions who have been "conditioned" by years of complex infrastructure management become incredulous about deployment of market data cloud platforms. "It can't be that easy!" is a sentence we often hear.

Flexibility and Innovation

Another problem with a monolithic architecture is that malfunctioning components can create global problems. For example, let's say that some relatively unimportant module starts a memory leak. Over the course of time, this can lead to negative consequences across the board. The behavior of the entire infrastructure slows to a crawl, taking everything temporarily offline. This is one of the reasons legacy became legacy. Changing components that are so interdependent is almost impossible. Next thing you know, it's 20 years later and you have your hands tied, your head in the sand, and a whole bunch of people whose existence depends on the preservation of that legacy planting their heels against any transformational change.

With microservices, you can bypass this type of issue entirely. Each microservice can be deployed and managed independently. You do not have to worry as much about version compatibility. You can reduce the infrastructure's failure footprint, since all modules are much less interdependent. You can decouple release cycles and increase business agility as fixes and updates can be developed and deployed

xignite

in hours. You can even let developers or vendors change and re-deploy software without fear of compromising the core application—which fuels innovation. Of course, a misbehaving service would still present a problem. But you can address it individually. And if it is not mission critical, you can choose not to prioritize a fix.

The Flip Side: Microservices Are Hard

Every good story has a catch and the one for microservices is that designing, building and deploying a true microservice architecture is not a walk in the park. It requires assimilating a huge body of knowledge that is very different from previous generations' software. It entails using and combining many different new technology components that traditional firms have little experience with. The whole devops process is completely different from before and managing operations effectively at scale and with low operating costs requires skills and tools mostly unknown to current ops teams. Actually, the effective use of microservices can be so complex that some firms choose to go back to monolithic architectures.

Of course, any IT team inside a large financial service firm will be excited to start working on these new technology stacks. But that does not mean that they will be able to move the needle on the firm's cloud-adoption efforts quickly. Actually, most likely, they will be consuming huge amounts of resources achieving very little--even maybe at the expense of keeping the legacy infrastructure humming.

Alternative: Hybrid Approach to Market Data In the Cloud

This is why the best approach is a hybrid approach to deploying market data in the cloud. Rather than building a whole new market data technology stack in the cloud as some very large firms have chosen to do, which would require significant maintenance of that technology stack, firms can choose to deploy battle-tested microservices like the ones we released here.

By deploying such components in your own cloud infrastructure, you get to instantly benefit from massive amounts of expertise and knowledge that would have required you years to acquire. You get to quickly learn about the right way to implement and manage microservices in the cloud. You can build that knowledge and expertise without sacrificing your resources and bandwidth to that learning.

Silicon Valley
1825 South Grant Street, Suite 100
San Mateo, CA 94402 USA

+1.866.965.7627 | +1 650.655-3700
xignite.com | @xignite