Deep Science Publishing
https://doi.org/10.70593/978-93-49910-78-2_4

Chapter 4: Designing cloud-native platforms for scalable financial services and Fintech innovation

4.1. Introduction

The financial industry has been largely dependent on its own private-developed monolithic systems. This traditional enterprise-wide architecture supported small business changes with long release cycles, many months to years of downtime, and a hard-to-understand code. The emerging agile-led fintech startups applying innovative solutions to upper-layer services on top of these legacy systems have outpaced the old-line financial institutions in boom and bust, leading to reduced customer loyalty, especially in risker retail and small business areas, and ultimately to the IPO or bankruptcy of some incumbent players. The incumbent banks have been forced to recreate their own versions of these fintech services with much more functionality, data, and regulation compliance; yet, styles and economics more on par with and compatible with these emerging competitors. The urgency, size, and costs of the changes required, and the likely added future legacy loading because of their achieving these goals through their traditional processes and tools, has led to increasing interest in two solutions that historically have supported these agile startups: cloud-native applications and services and the application of financial industry systems design best practices such as APIs on top of the new and traditional back-end, monolithic and emerging microservice, systems (Sharma & Chhillar, 2018; Hossain et al., 2019; Gupta & Gupta, 2020).

Both the cloud-native strategy and the best practices are covered in the related sections. The traditional banks have aligned on the controller stratagem. The foundation of their defense and attack has been initially nearly the same: building the new cloud apps just as the fintechs — old-style requirements-driven applications on top of an API pointing to the back-end legacy systems with various modernization accelerators or middleware. This monolith-on-cloud drive is not surprising, as these existing highly regulated service providers are reluctant to turn their customers' privileged data over to third parties and are still constrained by the weaknesses of their current legacy systems, processes, and tools (Yip, 2017; Sahu et al., 2021).

4.2. Understanding Cloud-Native Architecture

High-growth digital-native companies that are profitable, fast growing with awesome talent in the engineering and product space, also have modern technology enabling growth, innovation, and speed inside the company. Generally, these companies use cloud-native infrastructure that is designed from ground-up for the cloud with full realization of the possibilities and incentives that the cloud enables.

This architecture is not "just" cloud-hosting, nor is it just microservices. The set of technological principles and architectural patterns which are prevalent in these companies is broadly termed as cloud-native architecture. Creating and re-architecting systems as cloud-native is a design approach that needs to be adopted by a company.

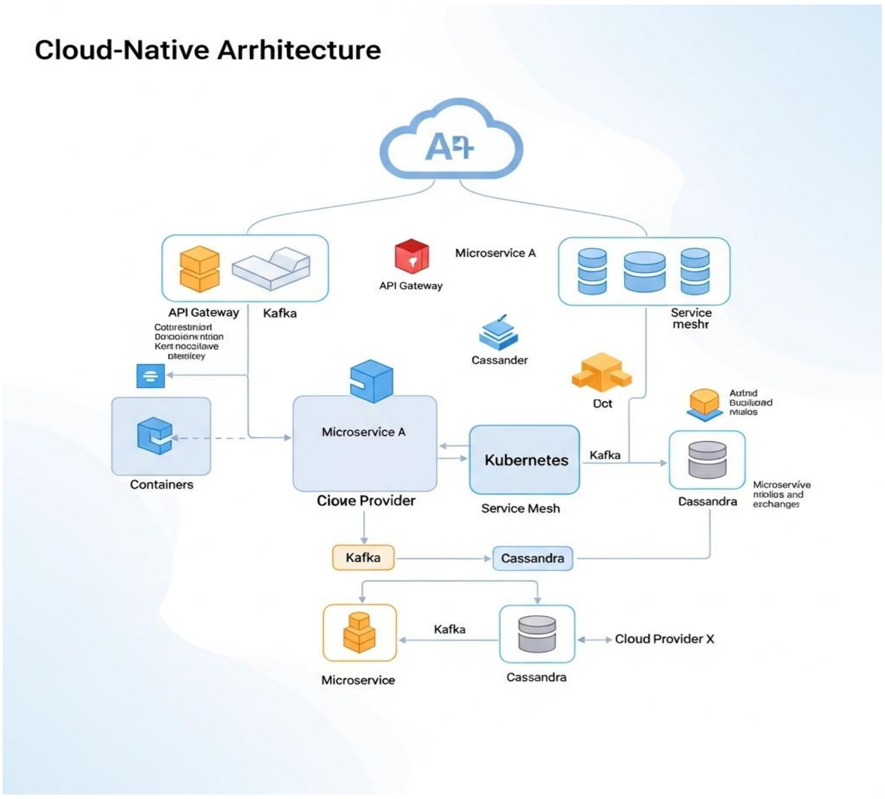

Fig 1: Diagram illustrating a cloud-native architecture.

Most enterprise companies have legacy systems that have been developed over decades and were once state-of-the-art in their design and deployment capabilities. However, if these enterprises want to grow as fast as their digital-native competitors, they need to innovate, innovate, innovate both in the products they develop for their customers and in the technology systems that enable and support innovation, speed, and scale. Cloud-native design enables that in a big way through re-architecting components of company systems inside an enterprise in a cloud-native way. These components then act as building blocks for agile, quickly deployable, easily scalable and extremely reliable company systems to deliver innovation and growth.

4.2.1. Definition and Principles

Cloud-native design refers to an approach in software configuration, deployment, operation, and maintenance that takes advantage of cutting-edge features, functionality, tools, resources, and technologies available in the cloud computing environment. Cloud-native components have unique and characteristic qualities that take advantage of cloud features that cannot be found in on-premises systems: clouds are elastic in nature, and cloud resources are universally and easily provisioned. These resources can also be released immediately when no longer used to improve cost-effectiveness. Clouds also provide multi-tenancy supporting architectures that offer improved performance, utilization of shared resources, and economics of scale. Cloud-native deployments have no single points of failure: they are highly distributive to accommodate potential failures in parts of the services.

A cloud-native design goes beyond configuration through existing vendor scripts to deploy containers in the cloud. Instead, cloud-native design recognizes the fundamental re-architecture of key services such as resilience, scalability, upgradeability, and security—not just from within single software systems but also in the way such systems interconnect with their supporting infrastructure and with other cloud services, whether internal or third party—into services that are not simply "better, faster, and cheaper" but are additionally more elastic and distributive on the data side, and more flexible and configurable on the operation side, that the cloud architecture offers. These fundamental principles, therefore, are all layered on the basis of both proper data architecture and correct operation-layering choices for cloud-native design.

4.2.2. Benefits of Cloud-Native Design

The definition of cloud-native refers to a set of capabilities that enable companies to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. This concept is tied to certain design models of the services and systems. First, cloud-native design favors a microservices model, structuring applications as a set of loosely coupled services. These services are deployed independently, embrace failure, and are observable. Together, the microservices implement complex business capabilities and allow for incremental, rapid development and deployment of customer-focused functionality moving at service speed. The second design principle has to do with the cloud as a runtime platform. Cloud-native systems take advantage of the capabilities of the cloud infrastructure. Cloud environments are elastic, meaning that they allow the application to easily scale up and down when demand changes. Cloud environments are also shared, facilitating multi-tenancy.

A series of advantages are relevant for companies looking to build an application for customers within the cloud environment. Scalability is the most evident one. Applications can efficiently handle an increase and decrease in load. A cloud-native design provides a limitlessly scalable structure in which overall system performance grows linearly with the addition of more machines. Fast time to market is another well-understood benefit of cloud-native design. A cloud-native design provides capabilities for deploying applications, including continuous integration and deployment pipelines, plus implementation and packaging standards. With a cloud-native approach, innovation happens many times a day. If there are problems with what is released, they can be immediately rolled back. Performance efficiency is also an essential property of cloud-native infrastructure. Cloud-native services can run on minimal resources and scale up seamlessly, taking advantage of the capabilities of public cloud providers.

4.3. The Role of Microservices in Financial Services

With the rapid advance of cloud-enabled fintech innovation, financial institutions must innovate at unprecedented speed, not only to seize new opportunities but also to fend off challenges from nontraditional players. Digital transformation done right can lead to huge benefits for financial institutions. We are witnessing a significant shift in the way financial services institutions are architecting their digital platforms from traditional monolithic architectures to cloud-native microservice architectures. In a monolithic architecture, the digital platform comprises a tightly integrated stack of technology components. In financial services, this is typically a core banking system—in its finest sense, a collection of proprietary or commercial platform components that have been integrated at significant effort and expense over a long period. In a microservices architecture, the digital platform instead comprises a collection of loosely integrated components called microservices, which can be developed deployed and managed independently. The role of microservices in financial services is to provide the rapidly-deployable, highly-reliable building blocks to deliver the business capabilities required to support the rapid, innovative delivery and operational model that financial services digital transformation demands.

The traditional financial services system architecture—typically a decades-old core banking system—was designed to support intra-day operations, borrowed from the principles underpinning batch-processing systems used in other sectors. Monolithic financial systems are costly to operate, difficult to enhance as requirements change and new fintech capabilities come along, and painfully slow to adapt to changing business needs. As financial services have become real-time services, firstly for retail banking, and more recently for capital markets, the architecture of the core banking system—and of other specialist systems that are used to support capital markets operations—needs to change. Being real-time means that not only do we have to support service availability, but also the individual business operations—and the interactions between them—are interleaved at much finer levels of granularity than with intra-day batch-based models.

4.3.1. Microservices vs. Monolithic Architectures

First, it is worth underscoring what we mean by a microservices architecture. In this section, we focus on microservices in the context of cloud-native platforms, which take primary advantage of containerization to support microservices. In this perspective, microservices are small services packaged into containers, which are very small and portable software units featuring everything needed for the service to run – including dependencies and configuration files. It is this extreme portability enabled by containers the reason why microservices can be deployed into the fine grain resources of a cloud-native platform and at a greater speed compared to alternatives. By packaged into containers, microservices rely also on a very volatile resource allocation and management method, the orchestration platform.

One example of a cloud-native architecture that encapsulates the philosophies of Agile and DevOps, besides featuring the deployment of microservices in containers in an orchestration platform is called Twelve-Factor App. The Twelve-Factor App makes a number of recommendations in order to develop microservices for the cloud-native environment. These recommendations include concerning factors such as coding, managing dependencies, configuration files, stages, processes and other apps, backing services, ports, processes, and release, as well as deploy. We will not go too deep into discussions about the Twelve-Factor App. Rather, our goal is to underline some of the differences between a microservices architecture and a monolithic architecture. Previous to that, we will address what we mean by monolithic architecture in our context, which is usually accomplished with cloud-enabled architectures.

4.3.2. Case Studies in Financial Services

Typically, banks are characterized as organizations that have been built over time, mixing processes and systems along with different stakeholders and time-to-market, that have been expanded for a broad range of services and products throughout the years. And as financial institutions grow, their IT architecture often becomes older and more complicated to maintain. In fact, the design of APIs can provide developer efficiency, but typically bring monolithic solutions, where the available integrations create a dependency between IT platform and financial institution. Consequently, a large part of the IT budget becomes an investment to maintain an old system, limiting investments for new solutions and to maintain the market pace along with fintech and other challengers.

First, we discuss the case of a traditional bank that reported the need for a breaking point to renew its architecture and internal processes, by describing how the decision to migrate to a microservices architecture was made. In this phase, the microservices design was driven mainly by the optimization of the available infrastructure without changing the approach of the old monolith approach that impacted the IT organization. Then, we describe a decision process of a traditional organization that instead implemented from the beginning a completely new architecture from scratch adopting a microservices approach. Finally, we present the experience of a fintech organization conceived from the beginning with a microservices architecture.

Monolith-first and monolith-now banks see their applications made of large applications packages that provide sometimes independently different features, more or less some application packages are directly shared among all locations. Large packages are often complex and need years to be developed and tested.

4.4. Scalability Challenges in Fintech

Distributed financial transaction processing systems are challenged by two factors. First, the rapidly accelerating growth of nearly every conceivable financial transaction related to products and services, coupled with governmental and industry regulation mandating enhanced levels of customer service and protection. Second, the economic principles of transactional banking that are driving profit-margins on traditional relatively low-risk financial services involving the handling transmission, conversion, and investment of huge amounts of funds — to all-time lows. This paper will describe the first part of the rapid growth of the scalability challenge. Successful financial transaction processing companies and their traditional banking rivals are using state-of-the-art electronic communications infrastructure and distribution concepts to develop appropriate alternatives to the low-margin traditional banking business of guaranteeing the safety of household funds, providing immediate liquidity, and relatively low-risk investment of surplus household funds.

These new alternative transaction systems must focus on instant transfers of large rather than small amounts of funds, quick reliable refunds for fraud, and high interest payments on funds in these systems. These upstarts can be financially viable only if they can invest heavily in technology — especially cutting-edge expert systems — and implementation, and grow to critical mass quickly. Otherwise, having no funds to invest in the scalability of their risk assessment and transaction processing systems, they will become uncompetitive and remain unprofitable. The above-specified problems are the very problems of quickly increasing transaction volumes, changing transaction patterns, and demand for reliability on which our previous research on financial transaction processing systems has focused.

4.4.1. Performance Bottlenecks

Scalability is the capability of a system or process to handle a growing amount of work or its potential to accommodate growth, while cloud-native is a distributed approach to building, deploying, and managing applications that takes full advantage of the cloud computing model. Software products or services are considered scalable if they can sustain an elevated volume of activity without suffering unacceptable degradation or it can be unexpectedly rationalized to handle traffic bursts or an expanded workload, while cloud-native solutions are inherently scalable by virtue of being distributed. Building an application to be cloud-native is separate and distinct from hosting it in the cloud. Hosting an application in the cloud may enable high availability, allowing for higher uptime than traditional infrastructure but does not mitigate the limitation of its monolithic nature.

A number of the non-functional quality attributes address scalability. A foundational principle of cloud-native systems design is “scale to zero.” When there is no demand for an application, there is no cost associated with it, and cloud-native applications can run like traditional applications. This principle is central to the serverless model. A system that scales cannot be inadvertently filled to capacity and refuse requests from users. To ensure that resources for handling requests are always available, the application must have some means of communicating to the infrastructure that request capacity must be increased. An elastic system is one that can automatically scale up and down in response to demand. As such, elastic systems govern themselves, although depending on the service, that governance may require some fine-tuning. This challenge is somewhat alleviated in a cloud-native environment because the application code no longer has to concern itself with specific resource management directives for instantiation and lifecycle of the system’s resources.

4.4.2. Load Balancing Strategies

This section discusses several strategies that can be applied to undertake load balancing. These include client-side load balancing and edge code, which relies on distribution of workload, server-side load balancing and reverse proxy that redirects the request to the backend servers using IP hashing and round robin strategies, database load balancing that redirects the request for the legitimate database, and CDN that improves latency by routing end-users requests through their nearest edge servers into the origin server.

Load balancing is necessary for any service provider to manage its performance and reliability regarding customers' requests. Load balancing regulates the distribution of workload to different resources, ensuring that the request is being entertained in a manageable manner, that the latency is at a minimum, and that there is reliability and availability. Load balancing is possible through one of these approaches: client-side load balancing, where the client takes the conscious decision to decide where to send the traffic; edge code, where some standard logic is being performed and decides where the traffic needs to be sent to; server-side load balancing, where some reverse proxy is intercepting the call to make the decision on where it is mapped; database load balancing, where all changes to the database are going through different mechanisms; and CDN, distributed networks that can be efficient to handle the traffic closest to the data source and decide where the traffic needs to be sent relative to how fast the content would be loaded.

Building a large-scale system or service comes with quite a few challenges, but managing resources and balancing them according to customer preference at a substantial amount of time is necessary in order to provide the consistency and desirable performance and reliability that the end-user expects. Load balancing is a service that can be either implemented using an external tool or can be built as a service, where the management and maintenance of rules defining where the request needs to be handled need to be managed well.

4.5. Data Management in Cloud-Native Financial Platforms

Cloud-native financial platforms deal with large amounts of transaction data. It's important to choose the right technology for storage and retrieval. Cloud-native systems have easy access to a lot of potential database options. The first consideration for any financial service application is for transactional safety. Sharded SQL stores provide strong transactional least common ancestor (LCA) guarantees across geographic boundaries, but it comes at a cost: these are relatively complex technologies that take time to learn and require some development effort to integrate and manage effectively.

They also usually have operational costs that exceed many NoSQL options, especially when you need to use a geographic span.

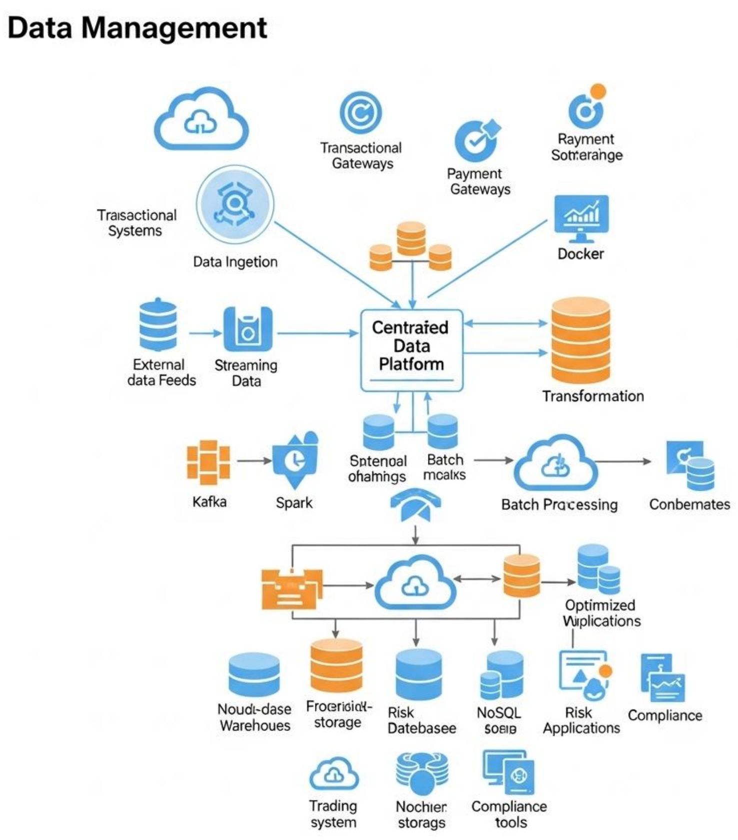

Fig 2 : Diagram illustrating data management in cloud-native financial platforms.

Eventually consistent distributed document stores or key-value stores can be lower-cost options assuming you can work with eventual consistency. Especially if you are serving a geographically close customer base where transactional enrichment is uncommon. All of these options provide a path to scale with low operational requirements since the cloud service providers take care of a lot of the backend management for you. Additionally, NoSQL solutions are more likely to work with several niche natural language processing or machine learning shortcomings. Running these workloads from multiple cloud regions is still more complicated and on a smaller scale might not be worth the effort. Data models that require multi-key queries that do not match a common key have a performance cost and may not always be as flexible comparing multiple options.

4.5.1. Database Choices

Data management is a vital aspect of finance in terms of security, persistence and availability. Financial processes produce a significant amount of data that need to be stored, queried and analyzed. Payment systems are responsible for various types of behavior, including the periodicity, quantity and value of transactions. These systems should also be able to track information about canceled and failed transactions, which follow a different pattern across users and have impacts on discussion points. Because large amounts of sensitive data may live in these databases, proper governance and compliance controls have to be in place, along with strong encryption methods for data in transit and at rest. In this section, we will discuss the main issues related to data management and data storage adopted by cloud-native financial platforms.

The term "database" means a storage system that provides a higher-level of abstraction in relation to the operating system file input and output systems. However, this term can be used to refer to NoSQL-based storage such as key-value stores, document stores, and object stores, as well as traditional SQL-based relational databases. Relational databases are built around the Structured Query Language to structure information in the form of rows and columns in tables. But due to the hierarchical and semi-structured nature of financial information, some financial services companies have opted for unstructured NoSQL approaches. Document storage is based on JSON objects; the customer profile is mapped inside a document. These documents usually contain a customer’s personal information and be dynamically updated with behavior data for purchases and sessions, along with any website error encountered.

Java has built-in support for using JDBC from other libraries to interface with SQL-based or NoSQL-based storage. In the case of NoSQL, there are dedicated libraries for each type of NoSQL storage. The Java language is quite popular in the enterprise, especially in financial services companies where it has been used for a long time to develop scalable solutions. However, microservices written using Java are usually heavier and consume a more significant amount of resources compared to microservices written in lighter technologies. These microservices interact with data storage through REST-based or message queue-based patterns.

4.5.2. Data Security and Compliance

Privacy and security regulations mandate that account and financial service data be stored and managed securely and in compliance with a number of global regulations. This puts additional pressure on the design of cloud-native platforms. Different regulations require different services to be exempt from the regulations, be part of them or to comply with them in a specific manner. Furthermore, compliance pressure also extends to the underlying Software as a Service capabilities offered by the cloud service provider, such as multi-tenancy and the fact that sensitive accounts are likely to be commingled with others in the proposed solutions. Coordinating the solutions designed by their financial services clients with their own to provide Security as a Service that meet security and compliance requirements is a priority for many cloud service platforms.

The requirements for addressing compliance often extend to infrastructure and network management, in the case of hybrid cloud implementations, especially when for example implementing a Zero Trust Architecture as well as access control. Choosing the required level of security and compliance for the cloud-native solution requires a risk-based security policy. Trade off decisions between risk levels, user experience and performance are the order of the day when designing cloud-native solutions dealing with security and compliance, and centers on the selection of the appropriate level or type of authentication authorization and auditing, that is required for achieving the expected level of security. The aim of any risk-based security policy is to blend together the needs of business assurance and compliance associated with security incidents and breaches with the expected cost profile of the solution.

4.6. DevOps Practices for Financial Services

There are several aspects to DevOps in the Financial Services context, which deals with legacy systems, secure information, and sometimes, sensitive data. These DevOps practices include continuous integration and deployment, monitoring and logging, reliability testing, collaboration and communication, security, and performance and scalability. First, in terms of deployment frequency and lead time for changes, the industry is not the same as the rest of the economy. Indeed, last year, the smallest e-commerce vendor had more visitors than the largest bank, including all its branches. Furthermore, its closing time is not Sunday at sunset. At the same time, often, changes to core functions in financial services happen several times a year, or even less. This leads to regulatory controls that create friction and slow down deployment, and in certain cases, problems with recovering from deployment failures.

In terms of DevOps lead time, the traditional model where information security reviewed every line of source code checked in is gone. However, the new model, where the developers check in code based on their good judgment, or a few automated tests, is also not tenable. Risks for organizations and customers cannot be ignored. Furthermore, organizations need not only detect whether code created vulnerabilities, they also need the right tooling and training to ensure that developers don’t re-introduce those vulnerabilities every time they deploy code. In fact, organizations have moved from trying to be the barrier of entry for vulnerabilities, to becoming their first line of defense, and laying the duty of care at the feet of developers. For this, organizations can consider offering security tools that developers want to use, especially if those tools also help streamline code production and deployment, help developers identify vulnerabilities, integrate within the tooling pipelines, and are cheap. And perhaps also help developers analyze the vulnerabilities before an external third-party security team tries to exploit them.

4.6.1. Continuous Integration and Deployment

Continuous integration (CI) and continuous deployment (CD) pipelines are automation setups for code changes in the source repositories corresponding to software components. CI/CD automates building processes and packaging newly developed code into deployable component files. It requires discipline to work synchronously, and a commitment of software developers, software testers, and others to ensure that they properly test working branches or trunk before changes are merged into it. Financial organizations implementing CI/CD pipelines need to focus on certain specific polling and resource-intensive challenges that affect these pipeline processes compared to supervised development in conventional banking systems. Hence, we must carefully look at how the CI/CD aims to tackle the complexity of overall resource usage and usability of cloud-native design in a fintech startup or in a banker IT.

A CI/CD pipeline for cloud-native design automates resource usage and sprawls control over time to offer the following advantages compared to manual controls. Continuous and automatic monitoring of all aspects including functional, performance, and operations requirements with emphasis on failover remediation detection from a particular QA environment. Support for technologies that automate building requirements for various software components or microservices according to the templates designed for them by teams of software engineers, testers, and operations that align with the overall architecture of the financial service app as envisioned by software architects. Automatic and intelligent decision-making for pipelines for a particular component to run, and the trigger conditions are met without waiting for human oversight. Finally, reports that offer a view of the entire organization’s microservices testing and deployment status related to specific releases in timeframes for the various fintech apps.

4.6.2. Monitoring and Logging

Monitoring and logging of deployed services are essential activities during the operation of a cloud-native platform that support feedback investigation, time-critical health alerts, performance insights, and data correlation for cyber alerts. Cloud-native architects choose suitable telemetry solutions supporting the business use and risk profiles of the design. Monitoring tools provide structure information such as health information on endpoints and databases, uptime status of services via configurable checking probes, circuit breaker status for handling failures for consuming services, and tracing data for application latency. Among open-source solutions, a popular metrics gathering tool integrated with a visualization platform provides configurable dashboards, dashboards templating, and alert notification capabilities through multiple channels. For specialized distributed request tracing, open-source projects provide up-to-date libraries and specifications for distributed request tracing backend repositories. On the commercial side, companies provide specialized monitoring solutions integrated with popular services while providing integration services for other common services.

The logging of services provides context-rich information beyond the well-defined metrics. Service logging sources to integrating need to identify in order for effective correlation of logs include request and response data, user identity and activity, service call logging level and component identifiers, and security error messages. With microservices, the correlation between logs from the multiple microservice calls needs to be designed upfront. For cloud-native services, a unified open-source solution for distributed tracing and logs is available. An open-source log library for microservices is also accessible. A managed service combines different open-source logging tools along with data monitoring and alerting capabilities. Similar to the monitoring solutions, there are a number of commercial solutions with data analysis capabilities integrated with alert triggers.

4.7. Regulatory Considerations in Cloud-Native Fintech

In creating cloud-native information security management systems and architectures for fintech applications, services, and infrastructure, we should consider regulatory statutes that apply to financial products and services, associated risk categories, and statutory requirements. Additionally, we should establish frameworks that can be used, both operationally and regulatory-expectationally, in risk management and risk mitigation.

Understanding Compliance Requirements

Most kinds of financial services and fintech platforms are subject to regulations; financial institutions must comply with myriad regulatory requirements, depending on jurisdiction, type of license, and financial services offered. Additionally, various branches of government have their own types of requirements. General principles of conduct, risk categories, and priorities are often construed as derived from these regulations. Data protection, consumer and client privacy and consent, operational stability, fraud detection and deterrence, communications security, protection of insider and insider threat risk, technology audits, risk-based security, and privacy training of employees are examples of risk categories generally derived from such regulations.

Risk Management Frameworks

Many fintech applications, services, and platforms must comply with regulations that mandate specific risk management frameworks and control implementations. As such, financial regulators also publish various regulatory guidance that dictate what framework to utilize, how to design and implement the control, and how to document compliance. Public, non-financial sector companies often must comply with the rules of the Securities and Exchange Commission or the doctrine of Sarbanes-Oxley.

4.7.1. Understanding Compliance Requirements

Compliance is one of the most complex and often drawn out aspects of any fintech business. Regulatory concerns vary by country, by offering, and by distribution. Regulations tend to have a long history, and each country tends to have built their independent set of regulations that have been amended along the way to reflect changing technological landscapes. Fintech businesses that operate in multiple geographies often find themselves grappling with these regulatory requirements, attempting to understand and adhere to them all. As fintech businesses grow and develop and begin to rely more heavily on third-party Cloud infrastructure, building for compliance needs to occur at every level of development. There are also a growing number of regulatory frameworks that govern how organizations using Cloud infrastructure need to manage risk.

Because fintechs interact with money movement in essentially every way that regular businesses do – from payroll to benefits to ecommerce to fraud detection – they are beholden to a multitude of compliance checks, not all of them directly related to managing finances. The primary set of regulations that impact fintech come from trusted industry regulators who oversee all financial services being offered. For banks in the U.S., it is the Office of the Comptroller of the Currency. For smaller institutions, it is either the Federal Reserve or the Federal Deposit Insurance Corporation. The Securities and Exchange Commission oversees those offering securities as a service. The Financial Industry Regulatory Authority oversees those in wealth management. The Consumer Financial Protection Bureau oversees anyone servicing consumer loans and debts. There are a relatively small number of settlors that oversee the financial services offerings, but the catch is that they apply to anyone offering these services, whether they own the financial employment or are simply acting as a facilitator.

4.7.2. Risk Management Frameworks

Risk management frameworks assist financial institutions in establishing policies and practical measures necessary to protect the organization and its customers. These frameworks address a wide range of potential risks, including information security risk management and incident response, business continuity and operational resilience, third-party risk assessment and management, risk management strategy, risk sensitivity, risk governance, risk monitoring, as well as capital requirements to protect an organization from any unexpected vulnerabilities. Financial institutions adopting a third-party service model, especially those utilizing cloud services, leverage the services of third-party cloud service providers that are critical for their service delivery. As a result, financial institutions could be subjected to increased risks, should the cloud service provider incur serious information security events, such as significant network outages and unavailability, information security breaches and serious data loss, or the serious loss of availability or integrity related to the use of other providers in their service supply chain. In order to protect against the potential losses from these events, financial institutions are required to adopt risk management frameworks that incorporate information security risk management and incidence response, business continuity and operational resilience, third-party risk assessment and management into the risk management framework, presenting serious challenges for the implementation of integrated risk management frameworks.

To verify the validity and effectiveness of the risk management framework and enable greater confidence in the overall integrated risk management framework, independent third-party assessments or audits of the critical cloud service provider's implemented risk management framework, including the controls supporting their risk management framework design, could be facilitated. Although other industry assessments could assist financial institutions in determining the level of risk involved in utilizing key services from cloud service providers, these standards do not contemplate all of the risk considerations for information security that may be necessary for financial institutions in accordance with the risk management framework. Given the highly dynamic environment of the cloud service sector, the Financial Supervisory Service intends to revise and supplement the current risk management framework and other supervisory guidelines in consultation with the financial supervision industry on a regular basis.

4.8. Innovation through Fintech Solutions

In modern life, custom product services can be adjusted for different market segments and customers, but they cannot respond to each customer. Digital finance solves this problem by creating services on finance product platforms and outsourcing the construction of the service to fintech companies. Just as individual users can declare their destination and create the best ride service for themselves, banks and payment institutions create situations that allow each customer to create their own loan product or payment service. But unlike driven cars, which solve customers' problems for only a few minutes, financial services with technologies provide continuous relationships for the duration of the transactions. Like a qualification test that continually separates good-talking customers from bad-dealing customers, the development of fintech solutions surrounds customers with many ways to interact, allow service providers to evaluate, and select qualified parties.

However, the big question mark is who specifies what services for the interactions. What fintech solutions can banks and payment companies open to request capital flows from customers? Can fintech companies propose combinations of variables that existing providers do not include in their business ranges? Even more fundamentally, who sells what to whom? The obvious answer is that the regulators will define what products banks can commercialize for which customers. However, this answer only repeats the bank-run dimension of the available model. This limitation is reasonable for transactions between individuals and banks. However, it does not make sense for services produced by transacting individuals. The market of finger-talking individuals is a vast pool of transaction creative ideas. Merchants play with interestary variables for all transaction steps within these dimensions. Instant payment for phone bills or bank account fund transfers can only come from the ways the transactions are modified by phones connecting every individual with the seller.

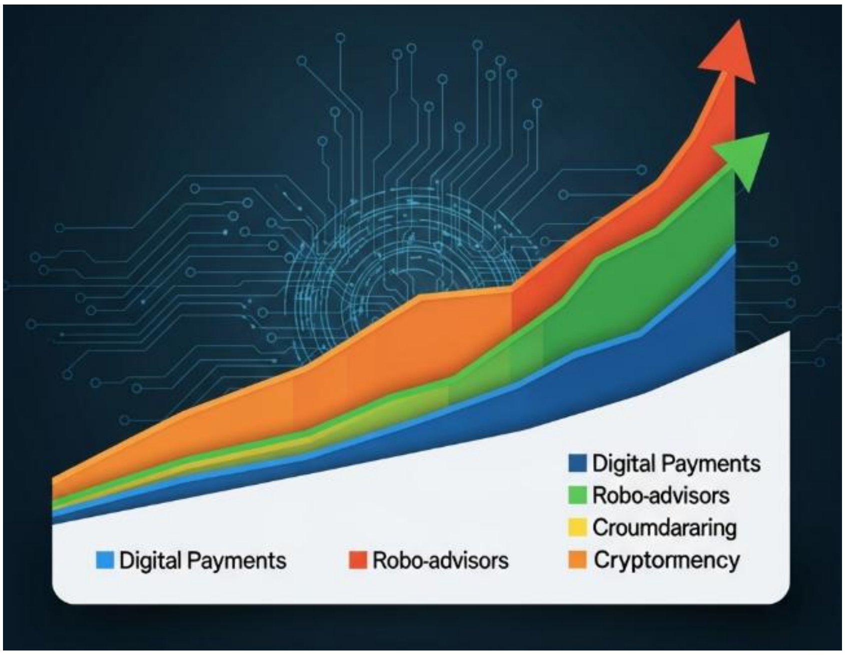

Fig : Graph illustrating innovation through Fintech solutions.

4.8.1. Emerging Technologies

Emerging technologies are driving innovation in financial services for both banks and new fintechs. These inventions include advanced data analytics, artificial intelligence, distributed ledger technology, natural language processing, and cloud computing. Although not new in the IT space, these technologies are now maturing and combining in different configurations to yield innovative solutions. And with the emergence of open banking standards, banks are increasingly exposed to fintech solutions that are more impactful than ever.

Open banking represents the best way to incorporate fintech products into a bank's existing suites of products and services. APIs allow financial institutions to unlock their data and let authorized third-parties provide optimized services to their customers. A bank can choose to bring fintech products and services into its own ecosystem via API integrations, or rely on third-party aggregators to act as the bridge between customer and fintech. Vendor platforms allow fintechs to build their solutions on top of existing financial infrastructure.

The value of technology is to help create and maintain competitive advantage. Banks may choose to innovate via established fintechs, who have laid the groundwork and experimental scars to become leaders in their areas of expertise. Some of them have already integrated emerging technologies into their solutions. By partnering with the right fintech organization, banks can significantly lessen the time required to implement a solution. These banks can then focus on their core competencies of customer service, risk management, and compliance. Banks' strength in these strategic differentiators will help facilitate the knowledge transfer required to actualize the solutions, allowing banks to innovate via fintech platforms without losing either party's sense of identity and purpose.

4.8.2. Customer-Centric Innovations

Contemplating the digitalization trends in general and in finance in particular, we find that there exist significant shifts in user expectations. As more retail services fundamentally shift online, it is likely that retail consumers will expect a functionally rich and context aware experience from online finance. Business clients will also expect personalized support with financial products and services much like the assistance for day-to-day operations provided by service providers or outsourcing partners. Millennial entrepreneurs are carrying their behavioral patterns from their lives into their businesses. This combination places digital platforms for finance into a uniquely powerful niche. It is essential to make Indians invest and save as much in proportion higher because of the backdrop that India has a large population of young people and is a consumption economy.

The infrastructure of payment transactions is essential for the economy so that it can function efficiently at low cost. A cheap source of frictionless transactions will allow companies to concentrate on what they do best: create products and content for consumers and drive the creation of new types of consumer relationships via digital interfaces. Establishing an open ecosystem of complementary dedicated service providers around existing customer bases with developed loyalty would help banks to better manage cost bases in new and challenging environments with disrupted margins. With a facility and benefit for everyone, banks will work together offering specific services to customers, depending on the financial acts they would mostly make: payments or savings, or loans. An open approach supported by standardized data sharing services is likely to host a number of market players that would target banks' existing customer bases to win higher market shares with attractive offers.

4.9. Case Studies of Successful Cloud-Native Financial Platforms

Cloud-native architecture is not a goal, but a means to achieve business objectives and provide capabilities to innovate and launch new products and services faster than competitors while doing so at lower costs. Hence, developing company strategies and cloud-native transformation roadmaps to ensure that business objectives are achieved is the first step, but different types of companies may pursue different strategies. This, in turn, creates diverse ecosystem opportunities for companies providing cloud-native, platform-based financial services and fintech innovations. In this section, we explore some concrete real-world examples of cloud-native financial services and their hosting platforms. Successful case studies are useful not only to minimize the risk of developing a company or platform strategy that is unsuccessful, but also to benchmark against industry leaders and, thus, enable companies planning a cloud-native transformation to anticipate the challenges in the transformation roadmap and avoid pitfalls on the journey. We explore the Leaders section of the Magic Quadrant of Global Leaders in Fintech for publicly available information about the macroeconomic context in which these companies operate. We also explore the lessons learned from the strategic and tactical failures of large enterprise banks and fintech start-ups attempting to build-and-run banks in a box. A case study analysis of global fintech leaders provides key insights into the capabilities required to drive strategic value and support success in development of cloud-native, platform-based innovations in financial services.

4.9.1. Global Leaders in Fintech

Many of the world's most successful leaders in fintech concentrate on designing financial platforms and services for specific phases of the life of their customers. For example, some companies have designed platforms aimed at customer needs in the present, or immediate timeframe, beyond just banking. Basically, these companies are in the business of being a digital wallet for a large percentage of customers in a financial ecosystem. In a similar vein, another company provides a new way for billions of consumers to pay over time, offering simple, affordable loans at checkout with no hidden fees, giving subprime consumers a chance to build their credit histories, and providing merchants more sales and less risk. In a nutshell, customers may choose to make their purchases with a traditional credit card, or opt for an installment loan, and at last count, a significant percentage of customers chose the latter option.

While others made a name in pure delivery of trading/stock services, one company and its apps went beyond the traditional banking model to become the leader in opening financial accounts for customers around the phrases of their life. By doing so, and executing consistently over a long period with their customers, this company managed to sign up a large number of accounts in just six quarters, and expand it to an even larger number of accounts four quarters after that. Its app commandeered a significant share of the peer-to-peer market in the U.S. With another payment solution, consumers can benefit from a simple, transparent, and easy-to-use service that helps budget their shopping. Selection is available through a diverse portfolio of brands in trendy fashion, beauty, and lifestyle, among others.

4.9.2. Lessons Learned from Failures

Some financial sector players proceeded with cloud-native transitions hesitantly and hastily. They failed, sputtering and splintering for different reasons—a litany of costly errors shared among executives and strategists in startups and megabanks alike. The motivation for many digital innovator wannabes was the fear of missing out. Leadership in established players could not help but see the massive financial and publicity winds directed at upstarts, including remarks from analysts who predicted a kind of death spiral for any traditional firm resistant to open data and cloud tools. The pressure to invest, innovate, and penetrate digital, economically scintillating, new space was impossible to resist.

In short order, private equity and venture capital firms poured money into dreaming amounts hopeful that incumbents were simply missing the giant, multiyear shift towards digital-first financial consumers. Bank leadership saw an opportunity to monetize digital expansion. Partnerships arose between early-stage startups and monolithic banks.

Restructuring began as internecine battles for budgets erupted between traditional product lines/enablers and digital innovation efforts, mounted as in-house startups. Unfortunately, the timelines for design, development, and implementation between the startup and the advance-squadrons of incumbency did not match. Failure rates for collaborations were monumental as banks had to wrest control back in-house and artists were fired. Too much time passed for startups to stay innovative, nimble, and focused on customer experience.

Time-to-sale and time-to-transact for resource allocation, development, and build in digital banking expanded as known challenges in globalization outstripped demand center design capabilities and resource allocation inside banks used for years in more solid times. Design authority was challenged as decisions 'flew' to the bank's upper echelons, where change was downright painful and impossible, symbolically and tangibly. Indecision, conflict avoidance, and gradualism set in as plans were put into action. The problems faced by both sides were manifest. Digital transformation merely became a bigger part of enterprise re-engineering efforts, without cloud-first, native-first direction and leadership needed for the needed strategy.

4.10. Future Trends in Cloud-Native Financial Services

Key future trends in cloud-native financial services include artificial intelligence and machine learning integration, which can leverage being data-centric rather than being app-centric. In addition, blockchain applications can optimize specific kinds of transactions by removing a third-party trusted intermediary. Other important considerations for the design of cloud-native financial services, especially for fintech innovation, are financial services as a human right, regulations as a design constraint, investor governance, IT service provider selection, and cryptocurrency technology.

AI and Machine Learning Integration

As the wide disparity of wealth and economic opportunity increases, questions of how to enable cheap, easy access to the financial services essential to individual and group financial resilience are on the minds of many economists and social planners. This question drives investment into cloud-native financial services that are based on vast amounts of high-quality data about interested and mobile users and accounts for successful pilot projects in microfinancing. The availability of huge data lakes, served by cloud-native financial services at scale with data-centric, multidimensional, and analytics pattern serviceability is motivating investment in AI core architecture and investing heavily in scalable data storage and application efficiency.

Blockchain Applications

Blockchain-specific distributed ledger technology work has a specific focus on peer-to-peer trust, which assumes an adversarial trust model, distributed authorization, shared state, unhackable data at rest, and durable uptime with native transaction granularity. The specific cloud-native financial services applications, especially for payments, remittances, and security transactions, focus on disintermediation, low transaction and service cost, arbitrary value encoding, and transaction privacy. CBDC-related cloud-native financial services vertical work originates from hostile nation-state actors and the realpolitik concerns of economic allies.

4.10.1. AI and Machine Learning Integration

One of the primary trends accelerating the transformation of traditional financial services into modern cloud-native platforms is the use of Artificial Intelligence and Machine Learning to create fully automated financial services or Fintech applications both at the private and enterprise levels. While Machine Learning has been an aspect of implementing financial technology for decades now, and vast funds of data have been made available by the services provided through connected networks, the opportunities to use ML for deep analysis of the data to automate how financial services work are just now being realized. What has changed in recent years to empower ML and AI are two key aspects: the availability of massive amounts of data, most especially about users' behavior and use of fintech services which guide AI systems to deliver better user experience and drive conversations, and the advance in the power of processing with cloud-native infrastructure technologies that help both store massive amounts of data that you can query and process through combined processing and storage capabilities or through massively parallel processing clusters built with orchestration.

These aspects, together with the vast increase of delivery of AI services by the big players in technology who are creating ML models that help build financial services or Fintech applications with better UX and conversational interfaces, augmented with AI interfaces, are making financial technology accessible, affordable, and user-friendly. In addition, AI can help finance companies that apply it improve their business processes and operations by for instance helping automate internal operations with intelligent bots that provide automated assistance to employees as they go through various workflows, augmenting Robotic Process Automation.

4.10.2. Blockchain Applications

Over the past fifteen years, blockchain has been touted as a revolutionary technology. It allows the creation of any kind of digital asset, such as digitalized securities or real estate, digital forms of value or different kinds of currencies, smart contracts that can trigger payments and streaming money, which enables concert tickets or electricity to be paid for automatically, and humanitarian aids and development cooperation. The group of companies who have stood out to date as fast movers in blockchain technology and services are the fintech platform vendors and some established high-to-financial-service organizations, especially in investment banking, who have engaged in ecosystem projects. After years of research and exploration in prototype development and proofs of concept in essentially all their business lines or domain spaces, they are positioning themselves to implement distributed ledger technology and cryptography at scale for a wider set of services and increase the partnership models to leverage niche players in other business areas, such as clearing and settlement, trade finance, regtech, mortgage-backed securities, full cash management services, notary and escrow services, and peer-to-peer lending. Blockchain technology creates new ways to think about trust, eliminating the need for a third party to facilitate transactions or provide verification. Especially in light of the current state of consumer trust in institutions, it opens up opportunities for financial services firms that want to play new roles in their customers' lives. Notably, blockchain can help strengthen the bank’s core by allowing it to support new clients, especially those operating in high-risks sectors. The financial services ecosystem is entering a new phase, where parties can rely on trusted transactions and data flows at record speed with confidence about the authenticity of their provenance.

4.11. Conclusion

Although cloud-native technology has disrupted banking service delivery, the financial services industry is yet to extract the full extent of opportunities. Innovative banks are quickly moving to put a 360-degree touch point strategy in place in order to achieve full customer engagement. They are making significant investments into integrating their service platform with other service platforms which they believe contribute to their customers' lifestyle. Their primary motive is to drive up service utilization and down acquisition costs by embedding their products and services within the fabric of their customers' financial and non-financial lives. Banks leverage the service expertise and customer engagement of external service providers by adopting an account aggregation approach to consumer financial services. If banking service delivery is a web, banks become the spiders while the non-banking providers of lifestyle services are treated as flies, placed within the web. Banks seek to entice customers to the bank’s part of the web so that the central depository for customer data can drive utilization of bank products and services that are part of the order flow when a customer makes a decision that impinges on their finances – making a purchase, transferring money, obtaining a loan, buying an insurance policy, saving for the future etc.

A customer centric order driven architecture is a paradigm shift in thinking about banking service delivery. Order driven architecture treats customers' financial lives as a continuous activity that involves making choices, some trivial while others are major life events. It is only when a customer is going through a life event that they become conscious of the consequences of the choice they have to make, which is that moment of truth; the bank must ensure that they have the necessary product and service on display by using data analysis to anticipate the customer’s financial needs during that life event and influence their choice of the institution used to fulfill their requirement.

References

A. J. Yip, "Cloud Computing for Financial Services: A Survey," IEEE Transactions on Cloud Computing, vol. 5, no. 3, pp. 456-469, 2017.

R. K. Gupta and S. K. Gupta, "Designing Cloud-Native Platforms for Scalable Financial Services," in 2020 IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), IEEE, 2020.

M. S. Hossain et al., "Cloud-Based Financial Services: Opportunities and Challenges," in 2019 IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), IEEE, 2019.

S. Sharma and R. S. Chhillar, "Scalable Financial Services on Cloud Platforms," in 2018 IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), IEEE, 2018.

P. K. Sahu et al., "Cloud-Native Architectures for Financial Services: A Review," in 2021 IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), IEEE, 2021.([sydneyacademics.com][2], [academia.edu][3])