Componentized Core Banking:

The next generation of composable banking processes built upon MongoDB

January 2023

MongoDB.

Summary 2

Chapter 1: The History of Core Banking 3

First Generation Core Banking: Digitization 3

Second Generation Core Banking: Product-Centric 4

Third Generation Core Banking: Customer-Centric 4

The Next Generation Core Banking: Process-Centric 5

Chapter 2: Potential Core Banking Solutions 6

The Core Banking Platform 6

The Core Banking Suite 7

Composable Ecosystems 7

Chapter 3: Building Blocks for a Composable Ecosystem 8

JSON 9

BIAN (Banking Industry Architecture Network) 9

MACH (Microservices, API-first, Cloud-native, Headless) 9

Data Domains 10

Chapter 4: Core Banking Use Cases 11

Temenos 11

Current 12

illimity 13

About the authors 15

Legal Notice 16

Summary

The financial services industry is in a phase of transition. Challenger banks and Fintechs are introducing new products at high speeds which requires established players to review their existing information landscapes. Often driven by regulatory and compliance changes, many systems, originally designed for never-changing processes, become the static throttle against innovation. Nothing more so than the so-called core banking systems that are the backend of all accounting and can reach back to the 1970s in their designs.

Thanks to the progress of software design and the emergence of better data infrastructure based on JSON, the next-generation of composable core banking processes is here, and it's built on a paradigm of a developer data platform.

In this whitepaper, explore the following chapters:

- Chapter 1: we provide a history of core banking and how the industry has evolved. We also explore some of the challenges that traditional core banking systems face, such as inflexibility and high costs.
- Chapter 2: we take a closer look at the current state of core banking and some of the solutions that are available. We also discuss some of the limitations of these solutions and how they may not be well-suited to the needs of modern banks.
- Chapter 3: we explore the building blocks that are necessary for a composable ecosystem. We focus on four key building blocks: JSON, BIAN, MACH, and Data Domains. We discuss how these technologies can be used together to create flexible and scalable core banking systems.
- Chapter 4: we provide real-world use cases for how MongoDB can be used to power core banking systems. These use cases demonstrate the flexibility and scalability of MongoDB and how it can help banks meet the changing needs of their customers.

Chapter 1: The History of Core Banking

Core banking, also known as the back-end IT solution, is the engine powering all banking functionalities to run day-to-day financial operations. First developed in the 1970s to support basic functionalities such as accounts, balances, positions, and movements, the core (centralized online real-time exchange) was a transformative technological innovation in financial banking institutions at the time. However, over the past 30 years, the core has evolved beyond four basic functionalities and has morphed into a technological monolith consisting of over 20 functionalities. Yet despite its additions, the technology foundation of the core continues to be based on 1970s technology.

According to McKinsey & Company, banks urgently need a modernized core platform based on a "flexible back end*". With rising banking disruptors tackling such legacy technology, and introducing new business models, incumbent banks must act quickly to innovate their core technology or risk losing market share and being left behind. To strategically move forward, it's worth looking at the past and understanding how we got here in the first place.

First Generation Core Banking: Digitization

Core banking's origins can be traced back to the 1950s, with the creation of the COBOL programming language. Roughly 20 years later, financial institutions began developing their *first computerized core banking system* based on COBOL and continued to do so until the 1990s. This development was a fundamental step towards digital transformation, as it led to an innovative way to perform tasks such as customer data management, transaction processing, record keeping, etc., all via a central computer.

However, despite the movement towards digitization, there were quite a number of difficulties. The technology was developed and kept in-house, which meant that it was only accessible to employees during business hours. The code was complex, data was stored in silos, and transactions were processed in batches at the end of the day (batch processing).

Factors such as time and cost efficiency, accessibility, and customer interaction weren't considered yet. Whenever development took place, IT costs grew substantially. Because these monolithic systems were deeply embedded and surrounded by ancillary support structures that were difficult to replace; updates also resulted in prolonged downtime.

Unfortunately, even today, "end-of-day" processing still takes place and is driven by the historic Control-M scheduler and batch processing.

Second Generation Core Banking: Product-Centric

With a “product-centric” approach from 1990 to 2005, banks began to outsource the development of core banking functionalities. Doing so allowed them to address specific products or groups of products by buying “commercial off-the-shelf” software instead of developing it in-house. The huge advantage of these kits was their broad range of specific functionalities.

Banking systems also began using certain subroutines and software modules to make their code more flexible. The banking interface became more interactive, and 24/7 access to banking services became possible. Yet these subroutines added complexity and resulted in a split between banks choosing to develop in-house and banks adapting to commercial solutions.

Whether developed in-house or commercially, the “product” optimization of core banking did not change the underlying data architecture. These systems continued to consist of siloed structures and the mainframe RDBMS (relational database management system) or Oracle PL/SQL remained the norm. The code was still quite complex and the inflexibility of the underlying data architecture was ever-present.

Third Generation Core Banking: Customer-Centric

Shifting away from a product-centric core banking practice, from 2005 to 2020, core banking became much more “customer-centric” with the end-user experience in focus. This shift resulted in fundamental changes to the core infrastructure. Banks abandoned the traditional silo approach by developing new software. This development focused on digital architecture models, such as service-oriented architecture (SOA) and application service providers (ASPs). For customer experience, this third-generation development meant increased accessibility, especially via digital graphic interfaces on the internet. Improvements surfaced in the user experience and development cycle. Updates took place on an annual or semi-annual basis, and version changes were performed over a weekend.

Despite this, the dilemma of static, complicated database technology persisted. Often the copying of data and the ETL (extract, transform, load) processes between databases led to a mixture of sources and critical data. These convoluted systems could only be traced back to “the core” on the mainframe or Oracle systems.

“We try every year to clean up our data landscape. We currently have 150 different source databases. After the next attempt to reduce, we will have 151.” - Architect at a G-SIB bank

This customer-centric iteration of core banking has resulted in tremendous improvements, but without a flexible database foundation, banks are merely scratching the surface. Incumbent banks fail to compete because they are stuck with a convoluted architecture that neo-banks have attacked and transformed with a process-centric mindset.

The Next Generation Core Banking: Process-Centric

Process-centric core banking meets both the needs of the banking institution and its customers. In other words, it is a *tour de force* of the prior product and customer-centric core banking generations.

Disrupters are building their foundation on a process-centric concept; this is illustrated in Figure 1 below. In the past, core banking systems were provided by only one vendor and each aspect of banking, such as customer accounts, internal bank operations, and other processes, were separated and handled by separate systems. Not only does such a siloed approach present challenges to both the bank and the customer; it also increases operational overhead for developers that find themselves building on top of these monolithic systems.

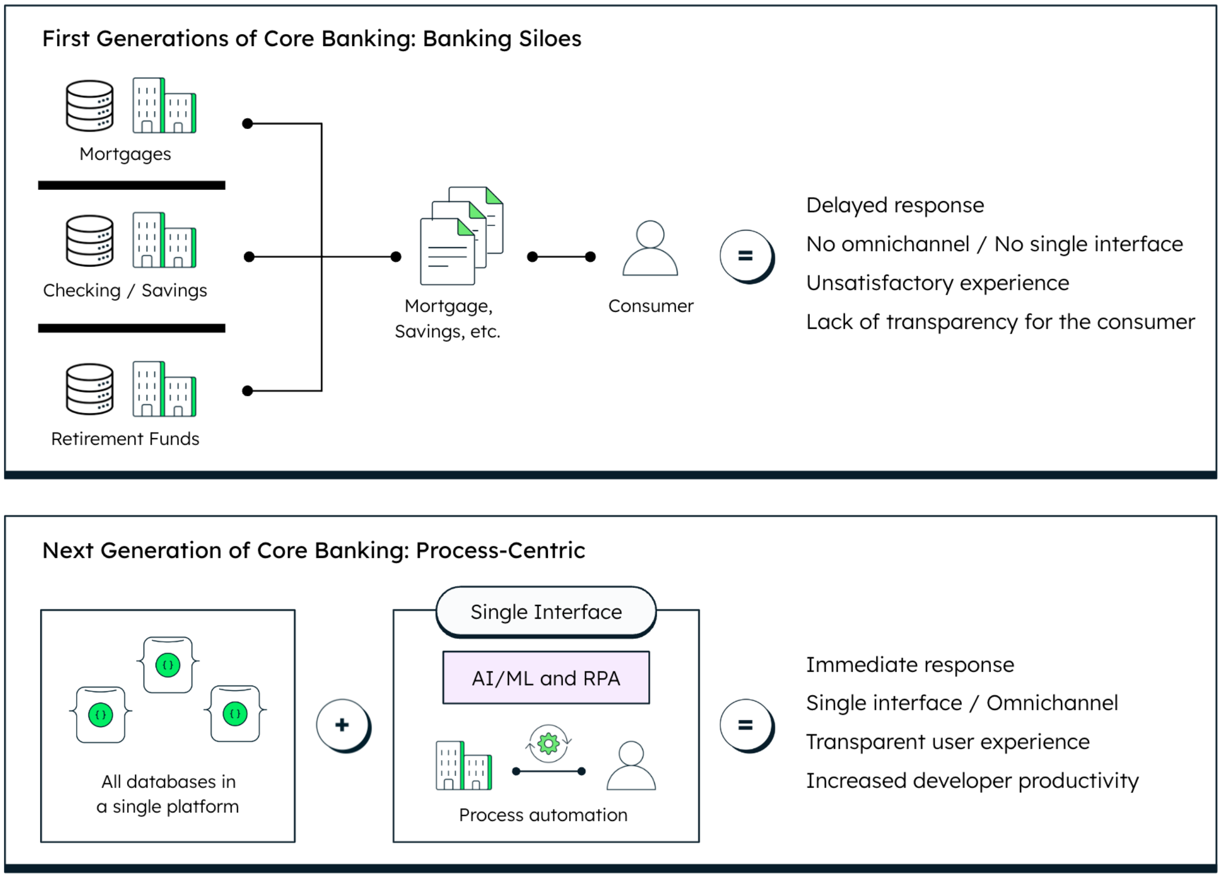
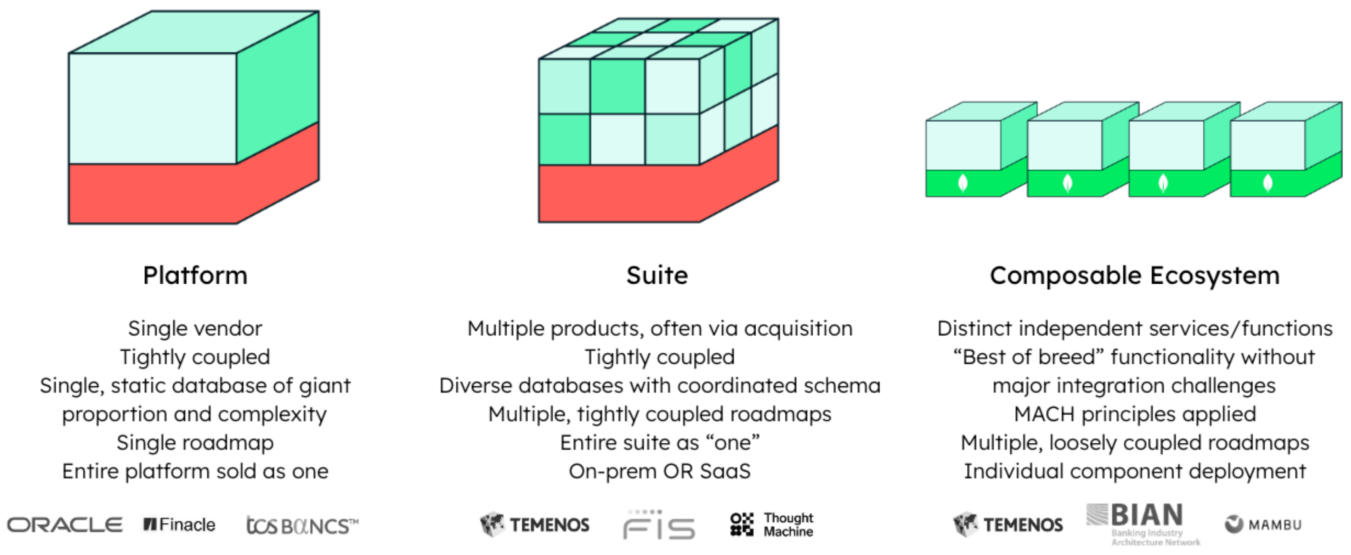

First Generations of Core Banking: Banking Siloes
Mortgages
Checking / Savings
Retirement Funds
Mortgage, Savings, etc.
Consumer
=
Delayed response
No omnichannel / No single interface
Unsatisfactory experience
Lack of transparency for the consumer
Next Generation of Core Banking: Process-Centric
All databases in a single platform
+
Single Interface
AI/ML and RPA
Process automation
=
Immediate response
Single interface / Omnichannel
Transparent user experience
Increased developer productivity

Figure 1: Evolution towards a process-centric core banking system: mortgage example

Only with a fundamental new start focused on “best-of-breed” and process-centric approaches – as it is presented by neo-banks – can you achieve this new start. These new core banking solutions enable financial services institutions to create efficient ecosystems that smoothly orchestrate interactions to offer an increasingly personalized customer experience, while allowing banks to launch innovative and competitive new products quickly and cost effectively.

Chapter 2: Potential Core Banking Solutions

As financial disruptors are growing their business and attracting customers built on the foundation of process-centric core banking, incumbent banks have hunted for solutions to tackle such legacy monoliths.



Platform
Single vendor
Tightly coupled
Single, static database of giant proportion and complexity
Single roadmap
Entire platform sold as one
ORACLE
Finacle
TCS BaNCS
Suite
Multiple products, often via acquisition
Tightly coupled
Diverse databases with coordinated schema
Multiple, tightly coupled roadmaps
Entire suite as “one”
On-prem OR SaaS
TEMENOS
FIS
Thought Machine
Composable Ecosystem
Distinct independent services/functions
“Best of breed” functionality without major integration challenges
MACH principles applied
Multiple, loosely coupled roadmaps
Individual component deployment
TEMENOS
BIAN
MAMBU

Figure 2: Core banking solutions: platform, suite and composable ecosystem

The Core Banking Platform

One such solution introduces a core banking “platform” seen on the left of Figure 2 above. Software companies such as Oracle, Finacle, and TCS BaNCS develop component-based solutions designed as native service-oriented architecture (SOA) platforms. Such platforms allow banks to break down each element of their architectural system so that it can be handled on its own without affecting existing components. Although this solution may present various modules, the foundation of the core consists of:

- A single vendor resulting in lock-in
- Multiple tightly coupled modules

- Single static and large database
- Single roadmap

Most detrimentally, however, is that the entire “platform” is sold as one to the client, which automatically removes the ability to adopt best-of-breed functionalities – outside the specified vendor – best fit for the banking institution.

The Core Banking Suite

As the core “platform” is inflexible and results in a single database responsible for all of the core functionalities, software companies have iterated upon and created banking suites. As seen in the middle of Figure 2, the suite consists of multiple databases; however, the underlying *technology* and *shape* of the core banking suite hasn’t changed. Instead, it has taken a “modular" approach by combining several pieces into one pre-set picture. Now you are left with:

- Multiple products often via acquisition
- Tightly coupled diverse databases with a coordinated schema
- Multiple, tightly coupled roadmaps
- On-premise or SaaS
- Entire suite as “one” a.k.a vendor lock-in

Although iterative and aimed at improvement, the results of both the platform and suite approaches showcase the fault of their legacy-based *database technology,* which simply cannot keep up with the requirements of modern day core banking.

Composable Ecosystems

Regulations are changing rapidly, advanced technologies are proliferating, and FinTechs are challenging traditional banking models, putting intense pressure on the financial services sector to innovate their data infrastructure. Finally, it seems that banks are ready to confront the serious limitations of their aging systems by adopting a different approach – one based on a ‘componentized’ model of the banking business as seen on the far right side of Figure 2. Such a composable ecosystem consists of:

- Distinct independent services/functions
- “Best of breed” functionality without major integration challenges
- Multiple, loosely coupled roadmaps
- Individual component deployment, no vendor lock-in

An industry that achieves componentization can specialize and develop more advanced individual components that can then be combined to deliver 'best-of-breed' products and services. Componentized industries are also better at exploiting new technologies and approaches, since changes are typically localized and affect only one or a few components without destabilizing them all.

Chapter 3: Building Blocks for a Composable Ecosystem

MongoDB provides financial institutions with the data foundation, what is called a developer data platform, which enables the building blocks to create a composable ecosystem.

The patterns for the composable ecosystem are derived out of four different areas:

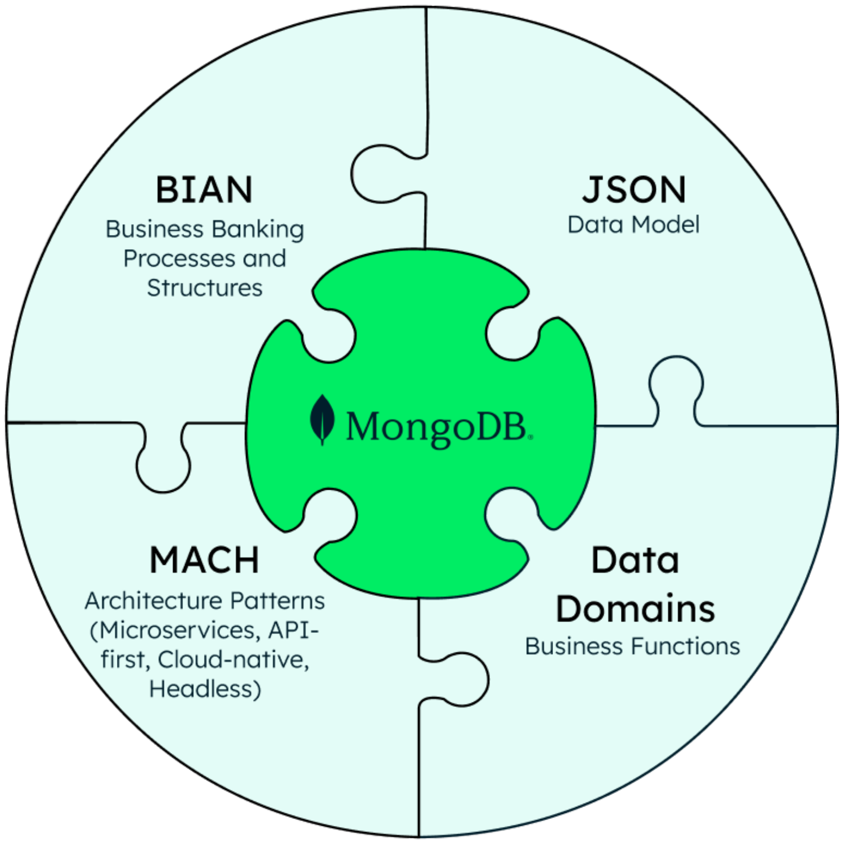

BIAN
Business Banking Processes and Structures
JSON
Data Model
MongoDB
MACH
Architecture Patterns (Microservices, API-first, Cloud-native, Headless)
Data Domains
Business Functions

Figure 3: MongoDB, the developer data platform for your core banking system

JSON

JSON is an extensible, easy-to-use, and polyglot data format that financial institutions increasingly depend on. According to McKinsey & Company, the use of JSON documents is the modern configurability standard in the next generation of core banking platforms. JSON's dominance as a data language is evidenced by the definition of JSON data models through standardization organizations such as SWIFT, ISO20022, and FIX.

Because JSON is limited in types, MongoDB invented BSON (Binary JSON), which stores data as JSON documents in a binary representation. Unlike most databases that store JSON data as primitive strings and numbers, the BSON encoding extends the JSON representation to include additional types such as integer, long, date, floating point, and decimal128. This makes it much easier for financial applications using MongoDB to process, sort, and compare data reliably.

By using MongoDB as your developer data platform, you are armed with a flexible JSON-based document model that increases developer productivity and is the back-end foundation for your next-generation core banking system.

BIAN (Banking Industry Architecture Network)

Since 2008, top banking application architects from organizations such as HSBC, Santander and JPMorgan Chase & Co., have worked to define the entire business logic of modern banking. The result is BIAN (Banking Industry Architecture Network), a standard defining a component business blueprint for banking. Developed specifically to address the problem of platform and suite complexity, BIAN enables banks to progressively componentize their business operations. This is done by promoting a best-of-breed concept to help financial services bring new services to market quickly and efficiently.

With BIAN providing a way for banks to achieve standardized business logic, MongoDB technology is not only the data backend for BIAN but also deliverers BIAN capabilities for banks by embodying MACH (Microservices, API-first, Cloud-native, Headless) principles.

MACH (Microservices, API-first, Cloud-native, Headless)

As one of the five global MACH enablers, MongoDB is the gold standard for architecting the data infrastructure for the future of core banking.

MACH (Microservices, API first, Cloud-native, Headless) codifies the design principles necessary for such component-based architectures. Microservices are defined as small components which own functionality and data for a given business domain. They communicate through APIs that define the contracts between the different domain teams. Typical application domains consist of hundreds or even thousands of microservices. Operating such systems in the traditional way would not be feasible; therefore, they can only be built with cloud-native technologies. In order for microservices to be able to be composed into applications, they have to expose all their capabilities and data through documented APIs. There is no need for user interfaces which are captured by them being headless.

Data Domains

With MongoDB's document model, you can bring data back into shape with the introduction of data domains versus convoluted table schema and ETL trains. In order to better manage financial architectures, data domains allow you to map business capabilities to your applications and data. Take for example, payments. A payment architecture that includes the merchant portal, fraud prevention and potentially an ad bidding platform. These business functions can map directly to data domains as seen in the payments example in Figure 4 below.

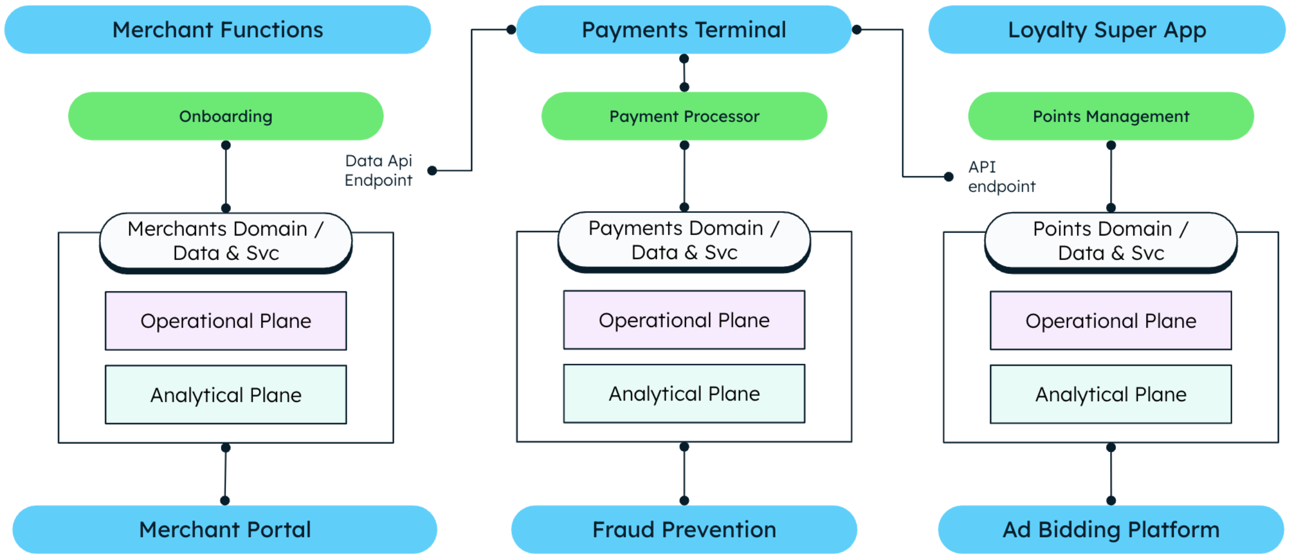


Figure 4: Data Domains applied to payments.

With MongoDB, financial institutions receive a component-based data infrastructure against which they can deliver microservices aligned to different business data domains – all independently evolvable, scalable, and isolated while making data easily shareable between applications.

JSON is the glue. The MACH principles and architecture template are the underpinnings of the BIAN service APIs and coreless banking process designs. MongoDB is the natural proven developer data platform for financial services.

Chapter 4: Core Banking Use Cases

Core banking use cases are a crucial aspect of the banking industry, as they enable banks to deliver a wide range of services to their customers. Such use cases are often tied to composable ecosystems, which are designed to facilitate collaboration, and innovation within the banking sector. As discussed, some of the leading composable ecosystem principles include MACH, BIAN, JSON, and Data-domains – all underpinned by MongoDB. These principles are used to create a flexible and interoperable environment where different systems can easily work together to provide a wide range of core banking services. For example, a core banking use case might embody the MACH principles by providing a scalable and modular platform that allows banks to easily add new services and features. Similarly, the use case might embody the BIAN principle by ensuring that the system is designed to support seamless integration with other systems, enabling banks to create a truly composable ecosystem.

In the following three use cases, MongoDB demonstrates its capabilities to provide financial institutions with a composable data ecosystem for the next-generation core banking platforms, encompassing JSON, MACH, BIAN, and Data Domain principles.

Temenos

"Implementing a good data model is a great start. Implementing a great database technology that uses that data model correctly, is *vital*. MongoDB is a really great fit for banking."

Tony Coleman, CTO of Temenos at MongoDB World 2022

Temenos is the world's largest financial services application provider, providing banking for more than 1.2 billion people and leading the way in banking software innovation. Temenos and MongoDB joined forces in 2019 to map out the path to data in a componentized world. Over the past few years, the two teams have collaborated on a number of new, innovative component-based services to enhance the Temenos product family.

Financial institutions can embed Temenos components, which deliver new functionality in their existing on-premises environments (or in their own environment in their cloud deployments) or through a full banking-as-a-service experience with Temenos T365, powered by MongoDB on various cloud platforms. Temenos embraces a cloud-first, microservices-based infrastructure built with MongoDB, giving customers flexibility while delivering significant performance improvements.

While thousands of banks rely on MongoDB for many parts of their operations, ranging from login management and online banking to risk and treasury management systems, Temenos' adoption of MongoDB is a milestone. It shows that there is significant value in moving from legacy database technology to a modern developer data platform with MongoDB. This allows faster innovation, eliminating technical debt along the way, and simplifying the landscape for financial institutions, their software vendors, and service providers.

If you'd like to learn about how Temenos and MongoDB are changing the core banking landscape, take a look at the following:

A New Era in Core Banking Data: How Temenos and MongoDB Change the Core Banking Landscape

From Core Banking to Componentized Banking: Temenos Transact Benchmark with MongoDB

Tony Coleman, Temenos and Boris Bialek, MongoDB | MongoDB World 2022

Current

"MongoDB gave us the flexibility to be agile with our data design and iterate quickly. The primary driver was the development velocity."

Trevor Marshall, CTO, Current

Current is a digital bank that was founded with the goal of providing its customers with a modern, convenient, and user-friendly banking experience. To achieve this, the company knew that it would need to build a robust, scalable, and flexible technology platform to power its services. Current decided to build its core technology ecosystem in-house, using MongoDB as the underlying database technology.

MongoDB as a trusted open-source database is well-suited to the needs of modern digital banks like Current. It offers a high degree of flexibility and scalability, allowing the company to easily add new features and services as needed. By building its composable core technology ecosystem on top of MongoDB, Current could quickly and easily develop a range of innovative banking services, including mobile banking and real-time transactions. This has allowed Current to provide its customers with a seamless and intuitive banking experience, setting it apart from traditional banks.

To learn more about how Current built its core banking technology on MongoDB, please refer to the following:

Current built its own banking tech. It's a secret weapon in a crowded field.

Next Generation Mobile Bank Current is Using MongoDB Atlas on Google Cloud to Make Financial Services Accessible and Affordable for All

illimity

illimity is a digital bank that was founded in 2017 with the aim of providing innovative banking services to business and consumers. As the first italian cloud-native bank, illimity has built its entire infrastructure on top of cloud computing technologies. This allows the bank to be agile and scalable, laying way to quickly and easily add new features and functionality to its platform. By using MongoDB as the foundation of its core banking architecture, illimity is able to take advantage of the flexibility and scalability of the cloud to provide its customers with the best possible banking experience. This has helped the bank stay ahead of the competition and continue to grow and thrive in the digital banking space.

"MongoDB has set itself apart from the field. They are an excellent partner in terms of research, development, and maturity."

Filipe Teixeria, CIO illimity

Today, MongoDB continues to support illimity's completely open and modular technological platform. Specifically, MongoDB Atlas' single view platform pulls in illimity's disparate data sources into one location. It can all be easily accessed, managed, and audited by staff or regulators, wherever they are. It saves money and regulatory headaches. Now teams can instantly see customer's interactions and provide the reliable service they expect. With MongoDB as their developer data platform to build a composable ecosystem, illimity is moving at incredible speed.

Interested to learn more about illimity and MongoDB, take a look at the following:

illimity & MongoDB: Revolutionary yet Reliable

illimity & MongoDB: rethink modern banking

Conclusion

Naturally, this paper could not discuss every single angle of modern core banking solutions. Many additional aspects like the approach for a migration and even the need for parallel operations – and how MongoDB resolves those challenges with ease – are subject for detailed discussions. The move towards a composable design and solution landscape is coming fast, and financial services institutions not heeding to this change may be challenged rather quickly by the competition. Consumer and professional clients alike expect fast-changing innovative experiences and the centricity of the user community is paramount.

To learn more about how the financial services industry is using next-generation data platforms such as MongoDB, take a look at the recent Forrester Study: What's Driving Next-Generation Data Platform Adoption in Financial Services.

Get in touch with the MongoDB team, to build your next-generation core banking platform on MongoDB's composable data ecosystem.

About the authors

Boris Bialek, Managing Director, Industry Solutions leads the industry solution practices at MongoDB and focuses on modernization and true innovation of FSI solutions. His experiences range from core banking, payments and cards to trading and risk & treasury. He is an industry expert in data technologies and recognized speaker and author. Before joining MongoDB, he worked for a lifetime at FIS, IBM, Dell and Compaq Computers. He obtained an MS degree from the Karlsruhe Institute of Technology.

Karolina Ruiz Rogelj is a cross-industry specialist for the Industry Solutions team. Coming from an interdisciplinary background in research, data analysis, and writing, she has a passion for translating industry knowledge into clear and compelling stories. She obtained a BSc degree in Computational Cognitive Science with minors in Linguistics and German from the University of California, Davis.

Ainhoa Mugica, is an industry specialist focusing on cross-industry collaboration. She is passionate about visual narratives, telling stories through the use of visual media. Ainhoa holds a BSc in Computer Science from the University of Newcastle and a Master of Digital Product Design and Direction from ELISAVA, Barcelona.

About MongoDB

MongoDB empowers innovators to unleash the power of software and data. Whether deployed in the cloud or on-premises, organizations use MongoDB for trading platforms, global payment data stores, digital end-to-end loan origination and servicing solutions, general ledger system of record, regulatory risk, treasury and many other back-office processes. At the core of our developer data platform is the most advanced cloud database service on the market, MongoDB Atlas, which can run in any cloud, or even across multiple clouds to get the best from each provider with no lock-in.

To learn more about MongoDB, visit MongoDB.com

Resources

For more information, please visit mongodb.com or contact us at sales@mongodb.com. Case Studies (mongodb.com/customers)
Presentations (mongodb.com/presentations)
Free Online Training (university.mongodb.com)
Webinars and Events (mongodb.com/events)
Documentation (docs.mongodb.com)
MongoDB Atlas database as a service for MongoDB (mongodb.com/cloud)
MongoDB Enterprise Download (mongodb.com/download)MongoDB Realm (mongodb.com/realm)

Legal Notice