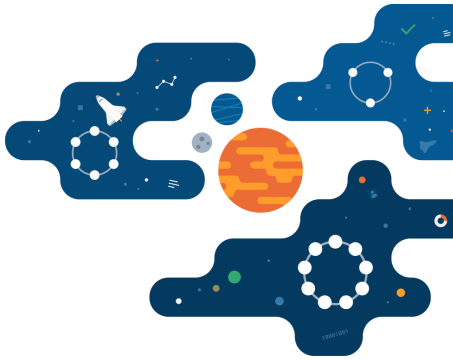
WHITEPAPER



Building a Risk-Mitigated Roadmap to Modernize Your Mainframe Investments

How to Cut MIPS and Boost Innovation Velocity

The role of IT is no longer only to keep the systems running; it's to deliver applications that provide competitive advantage–and that means getting the most out of a limited budget and innovating fast.

But mainframe maintenance and ever-increasing mainframe millions of instructions per second (MIPS) expenses continue to take a big bite out of budgets, in some cases consuming 40 percent or more.

Yet with so many digital initiatives focused around customer experience, and with mainframe systems being an essential touchpoint in customer processes, their utilization is increasing rather than falling.

DataStax

"Many mainframe-using organizations are evaluating their application portfolios in an effort to find workloads that can be moved to lower-cost platforms, thereby reducing consumption, or at least the growth, of mainframe MIPS"

HOW TO REDUCE THE COST OF IBM MAINFRAME COMPUTING, GARTNER

In contrast to ever-rising mainframe costs, the cost of running services on cloud platforms continues to drop every year. But expense isn't the only driver of modernization. Mainframe-based systems were often created decades ago to meet the demands of the time, not for the workload of tens or hundreds of thousands of concurrent requests from latency-sensitive applications. Users today expect real-time responsiveness across every channel; web, store, mobile–and while reliable, mainframes are increasingly challenged to deliver the performance that can now be achieved with elastic cloud platforms and modern databases.

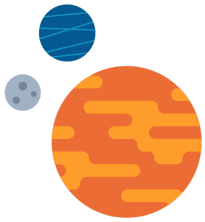
In speaking with CIOs at dozens of enterprises in transition, the same goals appear repeatedly:

- Dramatically reduce mainframe MIPS and associated tools expense.
- Meet more frequent and higher volume demand spikes without compromising performance or availability.
- Provide customers, agents, suppliers with incredibly responsive applications across every channel.
- Enable development teams to innovate free from the constraints of legacy systems.
- Shift mainframe workloads to the cloud–without ending up with "lock-in 2.0."
- Mitigate the risk of an aging workforce by moving toward modern technologies.

But how do enterprises morph core mainframe-based applications into microservices-based, ultra-high-throughput cloud applications, without creating risk? This paper provides an approach to modernization, garnered from DataStax field engineering experience, enabling shifting workloads to Cassandra and hybrid cloud at some of the world's largest telcos, banks, and other enterprises. This practical guide shares tips, technology pointers from IT leaders, and best practices to plot a path to modernization.

01

An Increasing Drag on IT Budgets and Velocity



Mainframe MIPS—Increasingly Hard to Manage and Predict

Managing the rising costs of mainframe MIPs is an increasing problem. Further, mainframe workloads have become less predictable. Batch-processing used to be the primary model–tightly controlled, and thus easy-to-forecast and factor into MIPS peak rolling average contract negotiations.

But now, with legacy mainframes being patched into fulfilling high-volume customer orders placed on a website, transactions teed up from a mobile device, processing returns via chatbot, and serving analytics data demands, workloads have become less simple to plan for. Compute demand is now highly variable, susceptible to seasonal surges or business-event driven demand spikes–which in risks underestimating or overestimating MIPS peaks, either way resulting in significantly more cost.

"The share of applications mainframes will be a part of is set to rise from 57-percent to 64-percent over a 12-month period."

FORRESTER CONSULTING

IT executives are still searching for ways to safely and cost-effectively move increasingly unpredictable workloads away from mainframe MIPS while minimizing incremental business risk during transition. The key is to take a phased approach, enabling a switchover of select workloads to a contemporary infrastructure in a controlled way.

Digital and CX Success Bottlenecked by Mainframes

These days, customer experience is everything; Gartner recently shared[1] that 75-percent of organizations have boosted their CX investments, creating a wave of new projects.

"23-percent of the mainframe workforce has been lost in the past five years, and 63-percent of those lost roles haven't been replaced, often due to skills shortages."

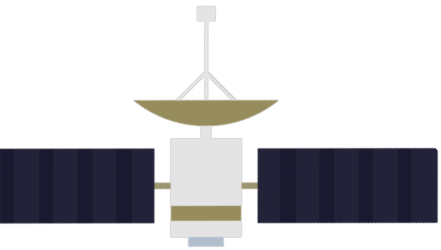
FORRESTER CONSULTING

[1] 2019 Customer Experience Innovation Survey, Gartner

But for enterprises with customer data and transactional processing occurring on a mainframe, it's estimated that nearly three-quarters of CX investments will still rely on mainframe processing to complete part of the transaction. And while surrounding mainframe apps with a digital layer can seem reasonable in the short term, beyond the MIPS issue, mainframes just aren't designed to play well as an API endpoint for modern applications. They often require kluges and scarce developer cycles to API-enable them.

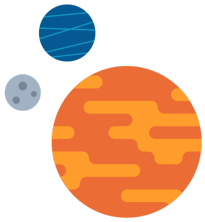
Change management is increasingly hard too, with the average mainframe workforce shrinking by nearly a quarter over the past five years, often due to retirement. Skills like JCL, COBOL, RPG, IMS DB, VSAM, IAM, QSAM, and numerous other job control languages, scripting, and database skills are increasingly difficult to find in the workforce. This loss of available talent also makes workload moves increasingly risky due to the loss of knowledge around application logic, and database schemas the longer that companies wait to begin transformation.

CX digital leaders require a way for their development teams to innovate, freed from the limitations of legacy mainframe code. But wholesale mainframe migrations that include application logic and database schema reworks are risky. To be successful, organizations require a gradual, risk-mitigated approach to shift core customer processing more gradually to modern databases, hybrid cloud platforms, and microservices-based infrastructure.



02

Four Key Technology Considerations for a Successful Mainframe Modernization



"Migrating away from the mainframe-based systems we had was challenging due to the huge number of customers we had to support, and availability demands. The phased approach we took implementing new services meant that we could switch over in a controlled way and avoid downtime risk. We saw huge benefits from the migration around delivering new functionality back to the business. If we had remained on the mainframe, it would have been extremely difficult to meet those requests in a timely manner, if at all."

RUNE BIRKEMOSE JAKOBSEN, DEVELOPMENT MANAGER, MOBILEPAY

Optimize For The Unknown

Until you get into a modernization project, it's hard to know the database model that will fit best, because the true scope of the application logic and data model becomes more evident as the project progresses. Thus, assuming that a relational schema is the only choice at the outset of the project can be a significant mistake. NoSQL databases handle a wide range of data types, using a much more adaptable model for handling data from diverse data sources. Better yet, a multi-model NoSQL database includes many data types and access methods in the same platform: tabular, wide-column, document-based, key-value, and graph data can all be accommodated by a flexible system like DataStax enterprise, with common analytical and search tools across all models.

Hedge Against Cloud Lock-in Risk

Most mainframe modernization efforts involve shifting workloads to the cloud, whether AWS, Azure, GCP, or the private cloud. However, the target database infrastructure must be cloud-agnostic to avoid vendor lock-in, which would otherwise make it prohibitively difficult to move in the event of hitting the kinds of technology constraints, performance and scalability issues, or usage-based pricing surprises that often occur later on during projects. Cross-platform NoSQL databases enable teams to quickly shift workloads to any public cloud, or even to deploy and run across multiple clouds. This allows ensuring maximum cloud vendor flexibility, and the ability to take advantage of the unique performance, pricing and scaling characteristics of each public cloud.

Plan To Meet The Highest Level Of Availability Expectations

Mainframes have a justified reputation for delivering mission-critical uptime over long periods. Many database platforms just aren't designed to provide the same level of service. Ensure that any candidate platform provides a highly resilient masterless architecture, rather than a master/slave, multi-master, or sharded architecture, to avoid a single point of failure or creating significant maintenance overhead.

Architect For QOS

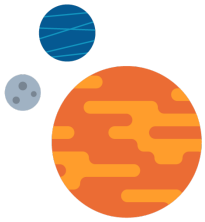
Mainframe apps are at the center of a massive array of business processes, and while workloads are increasingly volatile, mainframes still provide outstanding quality-of-service and throughput—for a price. Distributed NoSQL database architectures that elastically scale-out across public cloud, private cloud, or on-premises compute resources as demand grows provide the most cost-effective way of delivering mainframe level QoS without overprovisioning and overpaying.

"As scale-out, distributed infrastructures continue to evolve, organizations need to evaluate these with an open-mind so that they can generate the same throughput on these environments as they have obtained on the mainframes over the last several decades."

MAINFRAME MODERNIZATION: THE FOUR KEYS TO SUCCESS, GARTNER

03

Phased, Hybrid, Evolutionary. A Risk-Mitigated Modernization Plan that Works



Some of the world's largest enterprises like Capital One, Cisco, Comcast, Delta Airlines, eBay, Macy's, McDonald's, Safeway, Sony, and Walmart run DataStax Enterprise (DSE) for their mission-critical workloads.

Based on Apache Cassandra, DSE is distributed, multi-cloud ready, elastically scalable, highly available, fault-tolerant, and microservices-enabled. Advanced security, multi-model, search, analytics, and operational management capabilities make it the platform of choice for mainframe modernization projects at some of the largest telcos, biggest banks, and other Fortune 500 enterprises.

But what are the steps to build to shift from where you are today, to a modern database platform like DSE?

Think Incremental Workload Modernization

One of the biggest reasons mainframe modernization projects fail is an all-or-nothing approach. The good news is that DSE enables a gradual shift instead. For example, one large enterprise identified their mainframe-based Customer Accounts Management process as an ideal candidate, because it needed to be 100-percent available for customer self-service, with a swift response time, and integrated into a wide range of systems and processes across website, mobile, to call-center apps, so a microservices-based architecture was a significant benefit.

The data layer functionality was primarily Create, Read, Update, Delete (CRUD) operations, along with application logic around authorization, authentication, logging/auditing, and data transformations. Some processes access the data layer using OLTP, while significant batch operations provide information from other integrated systems.

"When a customer walks into a branch, we used to have to pull that information from the mainframe, which could take time. When we added search on top of Cassandra, the banker can then using the customer's phone number, name, address, or any information, can immediately pull the customer profile - or 360 view - information. And that's all accomplished in less than 50 milliseconds or so."

SESHU GUDDANTI, MANAGER OF DATA PLATFORMS AND AI/ML, US BANK

To phase in their mainframe modernization, they prioritized the Customer Accounts Management project ahead of migrating their mainframe-centric billing systems. The reason was that their legacy billing applications had poorly understood and written business logic around validation and reports of charges, credits, discounts, refunds, rebates, taxes, regulatory fees and subscriptions. While it was relatively simple to move these workloads to DSE from the database aspect, rewrite of the business logic from COBOL to a modern microservice was a tremendous effort. This proved to be the right call: not only did they get Customer Account Management modernization as a quick win, but the lessons learned were valuable for performing the billing upgrade in phase two.

Tip

In many cases, 90 percent of the transactions on a mainframe come from just a few dozen operations, and often the bulk of these are read-only. Identifying these high frequency, read-only requests can help mainframe modernization leaders quickly prioritize where to focus to drastically reduce MIPS and their associated costs, while at the same time building an innovation focused roadmap.

Architect for Smart Capacity Planning

By design, DSE can run distributed across any infrastructure on premises or in any cloud, providing maximum flexibility for performance and capacity planning. That's essential because the core of any mainframe transition is compute muscle. When DataStax Architects are estimating cluster sizes to host large total data volumes, they usually focus on data density per node and the performance that you need to achieve for your applications as the core concerns and can then direct the team to the most appropriate infrastructure to meet these needs.

The first step is to estimate the amount of data that applications need real-time access to, and understand what the performance requirements are. For example, is the expectation that certain reads/writes complete in milliseconds or seconds? As a distributed database, Cassandra has features that allow for excellent control over latency and performance. Cassandra's masterless distributed architecture allows the data to be stored and replicated closer to the customers who use it, resulting in reduced latency and a better user experience. Because DataStax is a distributed database, more storage and compute nodes can be added easily in order to achieve performance and resiliency goals, so that teams can quickly scale out the capacity as the data volume or velocity increases.

Evolve Your Schema Over Time

Best practice for an ideal migration project on paper would involve reworking the schema, but in the real world, this can sometimes create headaches that hinder the move. NoSQL databases provide a powerful advantage by enabling data model flexibility: teams can take an evolutionary approach by optimizing the data model for particular high-impact tables in phases. This strategy dramatically simplifies bulk data import and subsequent data synchronization with the mainframe that will often continue to be the system of record for some period. Datastax uses GraphQL to accelerate distributed and versioned schema definition and Striim CDC (Change Data Capture) and query-based pipelines to maintain consistency between RDBMS and distributed NoSQL systems.

Key Tips

- Consider selective denormalization to optimize what would normally be a frequent and/or heavy join.
- Use materialized views to facilitate easier querying, rekeying, or adding indexes to existing tables.
- Small lookup tables with rare/never changing values can be cached in the memory of the API servers or use DSE's in-memory capabilities to accelerate performance.
- Analyze the read and write workload of each table to inform data modeling, tuning compaction, and caching decisions.

Evolving the schema over time can ensure the data model works well in a distributed environment, ensuring that as the data grows, performance remains constant. Finally, evolving and testing models over multiple iterations provides the perfect starting point to begin analyzing production usage patterns and workloads in order to make empirical decisions about the production NoSQL data model.

Front-End Your Mainframe with Cassandra, then Transition

Most enterprises start their mainframe transition to microservices and cloud deployment backed by DSE by first effectively using it as a front-end cache, running in parallel with the existing mainframe. This enables them to immediately begin transitioning workloads to DSE and cloud infrastructure that would otherwise be directed to the mainframe, giving them fast query performance, and scale-out agility, while rapidly reducing MIPS usage without compromising reliability. Over time, teams can transition progressively their use from this cache architecture to handling the full application workload, and turn off redundant mainframe functionality entirely.

Multiple approaches are available to keep DSE and the mainframe in sync during the caching phase. Which one is most appropriate for a given application depends on performance requirements and the team's comfort level with introducing new architectural components:

- **DUAL-WRITE** changes to DSE and mainframe at the API level
- **PUSH** changes from the mainframe using a Change Data Capture tool
- **PULL** changes from the mainframe using a batch processing job in DSE Analytics
- **SYNC** changes between the mainframe and DSE using a batch processing job in DSE Analytics
- **QUEUE** changes in a persistent queue for two or more systems to consume (ESB), or write first to DSE and queue the change for other systems.

Take a Performance-Focused Approach to APIs and Application Logic

Many companies are now using microservices to re-architect existing applications because making the APIs between application components explicit enables easier Continuous Integration and Continuous Deployment, project agility, faster time-to-market, and higher developer productivity. In a ***recent survey*** by Red Hat, 69-percent of respondents indicated they are using microservices for both developing new applications and re-architecting existing ones.

"We wanted to implement a distributed database that would fit with our microservices-based application strategy and that would be able to handle the availability and scalability needs of the applications too. Cassandra matched this model perfectly, and the production support for DataStax Enterprise made a big difference."

RUNE BIRKEMOSE JAKOBSEN, DEVELOPMENT MANAGER, MOBILEPAY

In a mainframe modernization initiative, where performance is a priority, a smart strategy is to host the "hottest," most important APIs on independent groups of application servers and infrastructure, which can be equipped, scaled, and coded to meet the unique performance demands of those APIs. Good candidates to consider for this include APIs that handle:

- Caching/expiring lookup tables and other frequently referenced data
- Facilitate client-side joins
- Manage dual writes to multiple sources
- Perform write buffering

It's also helpful to plan what can be performed by the API service at a level above DSE to maximize performance. For example, client-side joins may be performed using reference tables that are cached and kept at the API layer. Technology extensions to Cassandra in DSE, such as DataStax Search and Graph, dramatically extend the capability of the operational data layer to perform functions of higher value to the microservice, especially when combined with performance driven client side optimizations.

Use Proven Best Practices for Tuning and Optimization

Quality-of-Service must always be top of mind for mainframe projects, and fortunately, DataStax Enterprise provides an array of options–and DataStax Field Engineering has a playbook for maximizing them. For example, trade off considerations for caching have to do with how frequently the lookup values may change, the acceptable cache invalidation strategy when that happens, the frequency with which the lookup values are accessed, and how many groups of APIs reference those lookup values. Lookup tables can be cached on the client, in the UI servers, the data-access API servers, forced in-memory by DSE or managed by the file system.

And often, because of the large number of tables involved in a transition, it is particularly important to understand the write workload imposed by each of the hosted tables and pay attention to how DataStax Enterprise is managing the in-memory and on-disk storage for those tables.

Create a Transitional Hybrid Cloud Plan

Hybrid cloud and multi-cloud architectures are becoming dominant in most enterprises, which store data and operate services across heterogeneous public clouds, private clouds, and on-premises infrastructure to provide the maximum flexibility and freedom. Using a data layer that supports these models is also a powerful strategy to de-risk a transition, rather than being completely reliant on a single cloud platform for success. DataStax recommends a six-step approach to identify dataflows and dependencies, and to determine how to move them to DataStax Enterprise, running on hybrid cloud infrastructure.

Identify And Document Existing Data Flows

Identify functionality in scope for transition, including key APIs, all data structure dependencies, and data in/data out flows. Because these systems have been in place for many years and there is often no single source of truth around business logic, discovery often involves collaborating with multiple groups across business and IT.

Factor In Legacy Workarounds And Old Kluges

It's common to find unexpected workarounds and data structures that are easy to dismiss as unimportant legacy details. However, further investigation often finds they serve vital functions even though they had to be implemented as a kluge at the time. It's essential to carefully document, factor them in, and use the opportunity to standardize and align them. The DataStax process ensures that these 'workarounds' are discovered at the schema definition phase and elevated to first level status.

Understand Data Structures That Interfaces Depend On

Once interfaces are identified, they should be grouped and prioritized based on criticality, and their dependent data structures (tables, consumed streams, etc.) should be fully documented.

Checkpoint

The result from these stages should be a clear, prioritized list of cohesive data structures that must be migrated and understanding of new microservices that depend on them, in order to begin building.

Build The Model, Establish Writes

With a clear perspective on interfaces, data flows, and data dependencies in hand, it's time to structure data in DataStax Enterprise, with decisions around the amount of denormalization, key and row design, and how the data will be distributed across cloud and on-premises platforms for best performance and efficiency. Using DataStax REST and GraphQL tooling, business objects are quickly expressed as DDL in the distributed persistence layer.

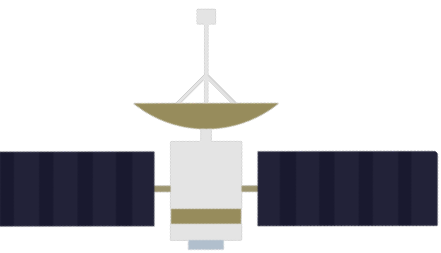
Building CDC- and query-based transformation pipelines, data moves in real-time between legacy systems and the new distributed data layer. This allows for zero-downtime migration to trusted transformational data architectures.

Migrate By Abstraction

With real-time data flows between legacy and transformed data models, applications that used to depend directly on the mainframe can now begin using new APIs, and related microservices. Legacy systems remain and continue to be populated, serving pre-transition applications and providing a fallback for the updated ones.

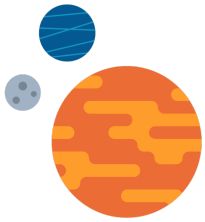
Gradually Phase-out Legacy

Once a collection of APIs that share common data structures have been fully migrated, and a period of production prove out is complete, bifurcated writes can be eliminated and the legacy mainframe APIs and dependent tables may be decommissioned and removed, providing a risk-avoided phased drawdown of mainframe MIPS. Alternatively, some organizations choose to partially decommission, or do so over a longer time-frame, depending on the criticality of the mainframe processes that are in-scope.



04

Next Steps



A mainframe modernization effort is never easy, but the benefits of undertaking one are manifold: lower costs, better agility, and an architecture that delivers improved customer experience today and in the future.

Fortunately, there's never been a better time to plan your transition, because not only is there now a proven playbook for success forged at some of the world's largest enterprises, but NoSQL database technology is now resilient, scalable, and flexible enough to make the shift, and combined with a cloud-neutral, hybrid approach, enables IT leaders to free finally free themselves from lock-in, and change the balance of power between them and their vendors.

To learn more about how DataStax products and our field engineering team can help, contact: ***info@datastax.com.***