How to Perfect Your Legacy Strategy: A Mainframe Modernization Case Study

Galen Silvestri, Senior Solutions Engineer
GigaSpaces
October 29th, 2020
Mainframes Are Here to Stay

Assembly of IBM 1401, Circa 1960
COBOL is Still Among Us

Reuters Report: COBOL Underpins Financial Industry

An aging programming language known as COBOL underpins much of the U.S. financial industry, but it has fallen out of favor among coders. This sets up a problem when systems run into glitches or need updates, and companies no longer have COBOL experts on hand.

- 43 PERCENT of banking systems are built on COBOL
- 80 PERCENT of in-person transactions use COBOL
- 95 PERCENT of ATM switches rely on COBOL code
- 220 BILLION lines of COBOL in use today
Mainframe Use is Growing

56% of infrastructure decision makers at these enterprises use the mainframe

46% predict an increased investment over the next two years

2019: Forrester’s “Tackling the Unsexy Challenge of Mainframe Modernization the Cloud Era Demands Connection to Modern DevOps Practices”
Challenges

• Missing Capacity. Extend mainframes? $$$

• Procure COBOL DEV’s? Is retraining this possible?

• Rip & Replace Mainframes? High Risk
# Typical Modernization Approaches

<table>
<thead>
<tr>
<th>#</th>
<th>Approach</th>
<th>Time</th>
<th>Effort</th>
<th>Cost</th>
<th>Overall Risk</th>
<th>Business Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Convert/ migrate and retire existing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>A. 3\textsuperscript{rd} Party solution</td>
<td>Medium</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td>B. in-house solution</td>
<td>High</td>
<td>High</td>
<td>High</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>2</td>
<td>Upgrade Existing Mainframe</td>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
</tr>
<tr>
<td>3</td>
<td>Augmenting technology on existing</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
</tr>
</tbody>
</table>

**Benefits of #3:**
- Path of least resistance – augment existing client solution
- Least disruption to existing Business and IT stakeholders – at all levels
- Reduces TCO and provides business value the quickest
Traditional Architecture

Result is that digital services pass thru these complex layers which impacts API performance.

Layers of ESB/SOA add complexity, requiring more people and specific skill sets.

Legacy systems of all types require special skills and a dwindling workforce.
Traditional Architecture

Result is that digital services pass thru these complex layers which impacts API performance.

How can we avoid Complex Architecture? And what are the benefits?

Legacy systems of all types require special skills and a dwindling workforce.
So What Does a Mainframe Modernization Strategy Look Like?

- **Capture your mainframe data**
  - Ex. DB2
  - Via: Change Data Capture (CDC)
  - Transfer to distributed in-memory data fabric

- **Leverage Extreme Transaction Processing**
  - ACID compliant
  - Fast analytics for business intelligence
  - MIPs offloading with Minimal latency

- **Optional Bi-Direction**
  - Expose COBOL applications or CICS API
  - Via microservice-based APIs
  - Integrate modernized applications
Proposed Mainframe Modernization Architecture

1. Continuously integrate and capture the data from a Mainframe (DB2) to GigaSpaces InsightEdge, a front end scalable & distributed data fabric.

2. Unifying extreme transaction processing and fast analytics for business intelligence and digital applications with minimal latency.

3. Automation of microservice-based APIs with modern SDKs triggering legacy COBOL or CICS applications.
Required Functionality

- **Mainframe offload**
  - eZ & quick access
  - Enrich via data lakes / warehouses

- **Microservices**
  - APIs
  - Event-driven analytics - Spark and BI

- **Translytics** Hybrid Transactional and Analytical Processing

- **Subsecond Data format agnostic ingestion.** IOPS M / sec with minimal ETL

- **Automatic Management**: Elastic Scale, DR, and Data tiers movement, indexing.

- **Deployment Anywhere** cloud, onPrem, hybrid, multi-cloud

- **Agility.** Support DevOps and modern data management
What Are The Benefits?

- **Reduce costs**
  - Smart Caching tier: offload mainframe MIPS
  - Infrastructure agnostic: Commodity Hardware and/or Cloud
- **Meet Availability SLAs**
  - Elastic scaling for peaks
  - Reduce overprovision
- **Modernize**
  - Innovate with modern apps
  - Future migration Journey
  - High-throughput, low-latency transactions
  - Fast data analytics & ML
- **Eliminate bottleneck** via on back-end applications
- **Faster time2market** for new modern services
  - Microservices architecture
  - Modern coding languages and frameworks (ie. Java and Spark)
  - Avoid new apps / use-cases addicted to on legacy
- **Easily migrate to Cloud**
  - Cloud native software that supports continuous migration
- **GigaSpaces WAN.** Efficient replication solution between remote sites,
  - Hybrid (MF + Gigaspace onPrem) ⇄ (GigaSpaces Cloud)
  - Reducing network overhead
  - Enforcing privacy regulations
Enterprises Require Flexible Implementation Paths

NEW DIGITAL APPLICATION (i.e. Open Banking)
- Capture Mainframe Data
- Develop GigaSpaces Microservices
- Minimal Additional Mainframe Load

EXISTING MAINFRAME APPLICATIONS
- Apps with Repeatable Queries (COBOL) e.g. get account details or payment information
- Cache data on GigaSpaces
- Significantly Reduce MIPS Load

Migrate
- Apps with High MIPS Consumption
  E.g. payment clearance or HNW
- Develop GigaSpaces Microservices
- Significantly Reduce MIPS Load

MIGRATE TO THE CLOUD
Real Life Case Study: PSA
Real-Time Pricing Engine

ABOUT PSA GROUPE
Groupe PSA is the second largest car manufacturer in Europe. PSA sold 3.5 million vehicles worldwide in 2019.

BEFORE GIGASPAACES
• WLTP regulation requires to calculate CO2 emission for every priced car. Compliance issues may lead to significant fines. Many car configurations are unique, but not all parts are significant for CO2 calculation.
• The mainframe pricing engine max capacity is 200 calculations per second. Demand is expected to reach 3000.

WHY DID REDIS FAIL
PSA tried to use Redis cache to offload queries from their mainframe. It failed because PSA needed multi-criteria queries but Redis was designed for a single index. Redis workarounds required replicating data footprint by 6X, with a major performance hit.

SOLUTION
• GigaSpaces Smart Cache was implemented with secondary indexes that allow multiple key queries.
• Digital applications query Smart Cache for CO2 calculation. Only if the result is not in the cache, Smart Cache will query the mainframe one time. After which future queries will be served from the cache.

RESULTS
• Redis footprint reduced by 6X by leveraging multiple indexes and avoiding unnecessary replication.
• Smart Cache response time reduced to 15-19 milliseconds vs 300 milliseconds mainframe response time.
• More than 95% of queries are served by a super fast cache, avoiding overloading the mainframe beyond its capacity.
PSA SmartCache & MF Architecture

PSA Digital Apps

Car configuration requests

SMARTCACHE

MULTI CRITERIA QUERIES
FAST AGGREGATIONS
SQL ENGINE
AI DRIVEN OPERATIONS

AUTONOMOUS SCALING

HOT DATA
RAM
PARTITION
PARTITION
PARTITION
PARTITION

MAINFRAME WLTP ENGINE

GIGASCAPES
Mainframe Modernization Simple Workflow
Mainframe is here to stay but you can now...

- **Reduce costs**: Smart Caching tier: offload mainframe MIPS

- **Meet Availability SLAs**: Elastic scaling

- **Modernize**

  - **Eliminate bottleneck** via on back-end applications

  - **Faster time2market** for new modern services

  - **Easily migrate to Cloud**

  - **GigaSpaces WAN**: Efficient multi cluster
Thank you!

For any questions, don’t hesitate to contact me:

galen.silvestri@gigaspaces.com