Utilitas Mathematica
ISSN 0315-3681 Volume 122, 2025

Transforming Legacy Mainframe Systems with the Strangler Pattern 2.0

Vishal Sharma
Vice President - Software Engineering, Broadridge Financial Services, New York, USA
vishalsharma.ic@gmail.com

Abstract

As technology evolved from the age of industrialization to the age of information, and finally to this current age of AI, the average lifecycle of technology shrank from 10 to 6, and now it is 3 years. Therefore, the definition of "legacy" has shifted. Applications developed before 2010, especially those built on mainframe or outdated technologies, are increasingly seen as candidates for modernization. Mainframe transformation has been a top priority in financial services, because of its' operational inefficiencies and high maintenance costs associated with mainframes when compared to modern distributed systems. Still, the majority of the core processing engines within financial services depend on Mainframe as the mission critical business rules are embedded in the code base. This embedded business rules complexity makes it challenging for transforming the system without disrupting the business continuity. The improved system moving away from monolith to capability driven design would enable significant reduction in time to market and the Opex cost. This paper illustrates an approach for transforming a legacy system leveraging Strangler Pattern, in conjunction with other off the shelf tool(s), test data management and other critical design aspects keeping business continuity in the center.

Keywords - Legacy Modernization, Mainframe, Capability Driven Architecture, Anti Corruption Layer, Strangler Fig Approach, Business Continuity.

1. Introduction

With the technology cycle reducing to three years; the term legacy has changed. Every application that has been developed before 2010 is a candidate for a case of modernization. The applications that are built on object-oriented languages, such as C++, Java, and others are still easier to modernize as they don't need transformation versus the applications built on mainframe and related dated technologies. Modernizing the legacy technologies, such as mainframe, that has intricate workloads, is a complex task, because it has organically grown over the years and has challenges in following three dimensions:

1.1 **People**: people who have built these technologies are slowly and gradually moving towards their retirement, therefore there is a fear of losing SME (Subject Matter Expert) knowledge.

1.2 **Process:** Mainframe applications were not designed to support distributed development methodologies such as Continuous Integration (CI) and Continuous

Deployment (CD), that are crucial for enabling faster release cycles, enhanced agility, and improved software quality. Additionally, Mainframe systems lack compatibility with modern source code management practices. Their procedural design complicates efforts to modularize and transform them. By modernizing the mainframe, organizations can introduce automation and continuous delivery pipelines that enable agile workflows. This integration allows development teams to make changes to applications faster, reducing release cycles and improving responsiveness to business needs. [10]

1.3 **Technology:** Mainframe systems, with their procedural architecture and tightly coupled codebases, present a significant challenge in adapting to the distributed system demands. Their inbuilt inflexibility makes modification and maintenance complex, resulting in technical debt, hinders scalability, integration, and overall technology agility. Organizations are currently struggling with prioritization, whether to run their businesses or invest in transformation. This transformation cannot just be a technology modernization. It has the potential to transform the way businesses are conducted, from improving time to market by at least 25% and reducing the OPEX cost by 30%.

Seeing the volatility in the market, organizations are not ready to commit for 5-7 years of transformation without yielding results every one or two quarters. Therefore, the transformation strategy must ensure "Business Continuity" and must be bite-sized. These bite sized transformed functionalities will start providing return on investment sooner, rather than waiting for the entire transformation to be completed. The following sections will provide options about the Mainframe Transformation using Strangler Approach.

2. Literature Review

Mainframe Transformation has been a focal point of the financial industry. Many studies highlighted the operational inefficiencies and high maintenance costs of mainframes in comparison to distributed systems.

Tata Consultancy Services' Global Survey from Feb,11,2021 [1], revealed that more than 70% of CxOs of global companies consider mainframe or legacy modernization a strategic business priority for the next three years. The study, conducted in collaboration with AWS, surveyed

211 CxOs and senior decision makers from companies that currently have legacy or mainframe applications. It highlights noteworthy trends on how companies view the future of legacy and mainframe applications in their digital transformation as bottlenecks and challenges in migrating those applications, and the preference for microservices based technologies.

The study published [2] by Paramatix went even further to explain about the outcomes, if companies delay their transformation plan. It starts by pointing out something companies already feel: legacy systems are expensive to maintain. These old platforms eat up huge chunks of the OpEx budget just to keep the lights on, leaving very little room for innovation. And it's not always about money. They come with a

growing list of risks—security vulnerabilities, compliance issues, and integration challenges with modern tools. One of the most striking points is how much these outdated systems slow companies down. They make it harder to adapt, to scale, or to roll out new features quickly.

The study [3] by Futurum Group explains that mainframe modernization begins with application modernization: focusing on replatforming critical mainframe tasks to execute on public cloud, while maintaining the business logic in existing Mainframe implementation. This approach enhances agility, reduces costs, promotes cloud integration, and helps attract tech talent—all without rewriting core systems. The core point that is not evaluated as part of this study, is how to transform the business. It can make an enterprise look good from the outside in, but internally, it'll still be tied to old business processes. This application "modernization" approach not "transformation" will be good for data presentation and/or data capture, not for true business efficiency. Therefore, a true transformation of any Mainframe application is a vision of evicting Mainframe, while delivering business value in process, Creating a System of Records, maintaining better, cleaner data, and reducing process bottlenecks.

Therefore, for a true business transformation, it should always be a combination of Rebuild and Replace, from the 7Rs [4] of Gartner. Here, Rebuild allows us to redesign or rewrite the application component from scratch while preserving its scope and specifications; therefore, the Stangler approach can be followed. And ultimately to replace via eliminating the former application component altogether and replacing it, considering new discovered requirements and future needs at the same time.

3. Research Gaps and Strategic Innovations

This draft reveals a critical gap in our understanding of Cost Management and Total Cost of Ownership (TCO), particularly concerning the Bell Curve for Capability Management.

Effective cost management is vital during periods of business continuity, as organizations must successfully navigate the coexistence of two systems over an extended timeframe.

This communication underscores the importance of maintaining business continuity and maximizing value throughout the entire transformation cycle, which can stretch across multiple years. Furthermore, should a pause become necessary during this transformation process, it can be strategically managed through the approach outlined above, ensuring minimal disruption and a clear path forward. By addressing these gaps, we can strengthen our strategies and enhance overall effectiveness in driving sustainable change.

4. Transformation Approaches:

There are different approaches to transform the legacy systems.

4.1 **Top down**: A top-down approach to system transformation is a methodology where the overall system's target state is first defined, and then progressively broken down into smaller, more detailed sub-systems and components. This approach starts with a high-level overview and gradually refines it with more specific details, working from the general to the specific.

This approach is an excellent approach where the business functionality is exactly known along with the business rules. Each and every aspect of the product is well documented that could be then converted into requirements for the new system(s). The challenge with this approach is, it's highly unlikely anyone can fully document the functionality that is buried inside the legacy code that has organically grown over decades therefore, there is a high chance that the target system or application built as a result of this approach will have significant capability gaps.

4.2 **Bottom up:** The bottom-up approach to system transformation starts by focusing on individual, self-contained components, developing and testing them in isolation, and then integrating them to build larger subsystems until the entire system is complete. There are challenges with bottom up approach such as difficulty seeing the big picture, rather focus is on individual modules that can obscure the overall system requirements. There are chances of potential for strategic misalignment due to lack of a clear top-down vision. Integration complexities and potential for flaws will also bring integration challenges for bringing modules together later in the development cycle.

4.3 **Meet in the middle:** The "meet in the middle" approach, is a hybrid transformation strategy that combines elements of top-down and bottom-up methodologies. It involves defining broad architectural guidelines (top-down) while simultaneously developing individual components (bottom-up). These two development streams then iteratively synchronize to refine the design and ensure alignment. A comprehensive way to start the system transformation is to start with building the target state capability design.

As Mainframe Legacy systems are majorly organically grown and with very limited availability of SMEs, it is highly recommended to follow a “Meet in the Middle” approach.

5. Meet in the Middle Approach

The following steps (Figure 1) are performed to start the “Meet in the Middle” approach:

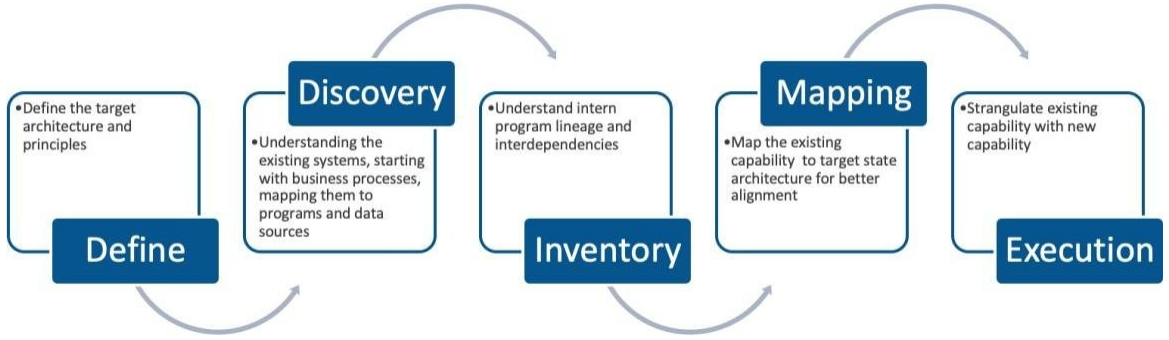


Figure 1. Stages of “Meet in the Middle” Approach

Define: In this step, the true north star system is designed. The capabilities are formulated, named along with sub capabilities and are also documented along with their Objectives and Key Results. This step a clear vision for the desired future state of business capabilities is established, aligned with organisational goals. After

establishing the business capabilities, establish a detailed capability model, mapping capabilities to business drivers. Capability Roadmap is also defined within this step along with the business value to be delivered in every increment of the business value, it could be quantifiable or even qualifiable. But the business value of every increment of Business Capability must be defined, documented and re- evaluated. These capabilities are horizontal in nature, in contrast to domains that are more vertical in nature.

The second part of this step is to define the data domains, and its owners. This is important to identify first and ensure there is no duplication of data or system of records (SoRs). There could be multiple authoritative data sources (ADS) but only one SoR, the source of truth. This will ensure data integrity and data democratization.

Discovery: This involves documenting the top down process partnering with the business and tech operation team along with the product. Mapping these processes to true Cobol programs and ultimately to VSAM files or DB2 tables. In this step it is recommended to document the process for the entirety of the system within the scope of transformation.

Inventory: During this step, all the capabilities identified are documented, in one process flow. Add the lineage, and dependencies between applications within a system. Data dependencies between applications, complexity factor provided to an application within a system. Additionally, logical boundaries are defined to distinguish individual applications within a large monolithic system, setting the stage for modularization and eventual decoupling.

Mapping: This is a critical step in the "meet in the middle" approach, where the defined target state capabilities are meticulously mapped to the existing functions. of the current application(s). After this mapping is accomplished, the capabilities are then prioritised for implementation using business value assessment and the functions are then phased out through a controlled strangulation process.

Execution: Once the north bound capabilities are identified, how do the execution start and from where? To answer this question, an exercise is performed to identify the highest value and lowest risk sub capability to be transformed first. To identify the business value and risk association, a “Business Value Assessment” methodology is being used. The Business Value Assessment Technique is a practical way for organizations to figure out which applications must be transformed first vs others . It uses a simple matrix that looks at both the potential value and the risks of each option. This helps the transformation team to make smarter, more informed decisions by clearly showing which app transform will bring the most benefit with the least risk.

5.1 Business Value Assessment - The Business Value Assessment Technique is a scientific approach used to evaluate the potential business value of different capabilities delivered against risk, timelines, and magnitude. This technique allows decision-makers to make an informed decision before embarking on a long execution state (Figure 2).

For each capability, assign scores based on revenue and risk if not transformed. Multiply each score by their corresponding weighted to get a weighted score. Weighted score of overall value and risk assessment.

The diagram below depicts a business value assessment outcome across different capabilities. The size of a circle denotes the magnitude of the capability. The colour depicts the running state and the location on the graph provides the information of risk versus value [5].

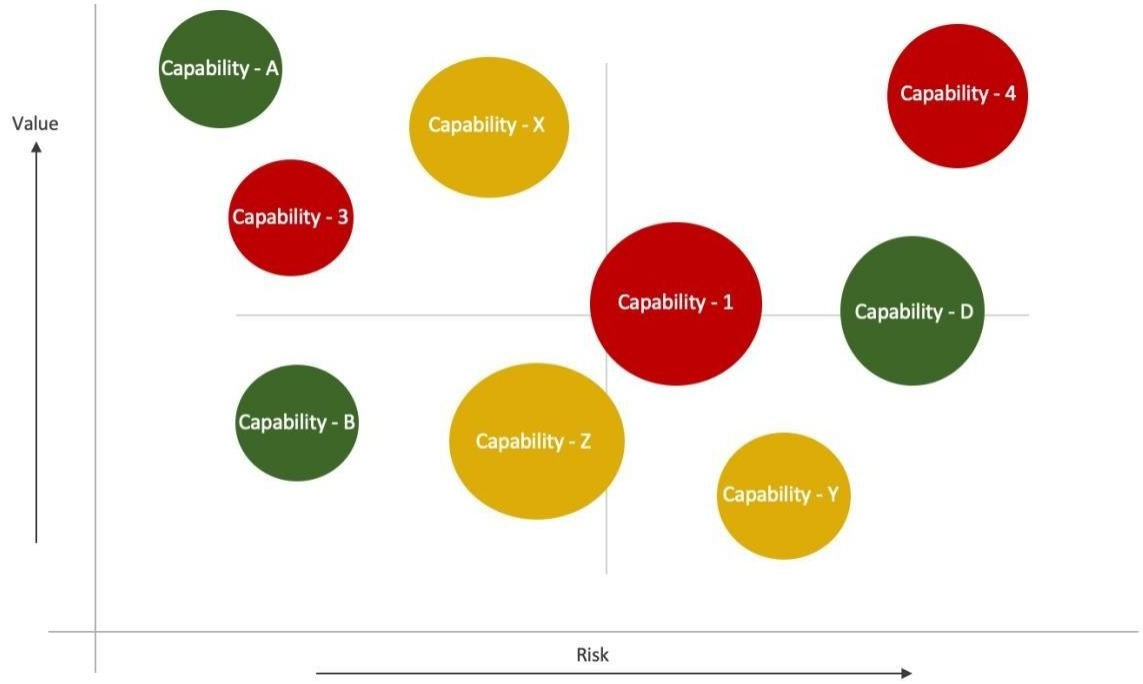


Figure 2. Business Value Assessment Chart [5]

In the above diagram, the stakeholders will pick Capability A because it provides maximum value with least risk, followed by Capability X and then Capability 3.

5.2 Implementation

Once the application is identified to be transformed, the next and most crucial phase starts of controlled Strangling a Mainframe Application within a monolithic code base (Figure 3).

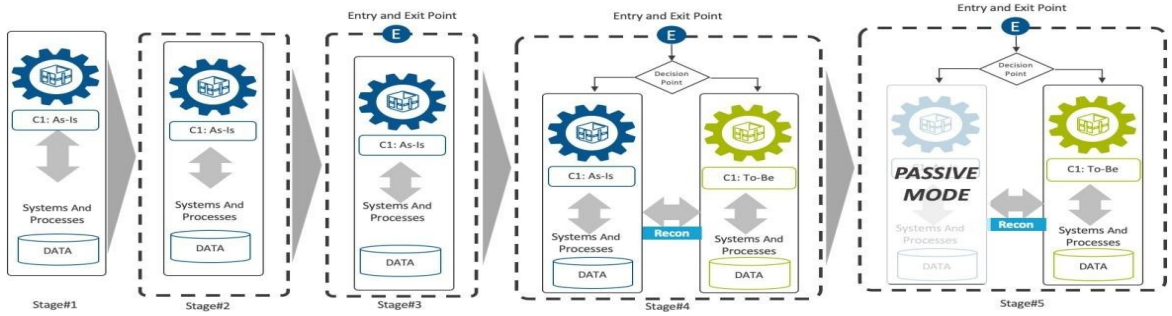


Figure 3. Strangler 2.0 Approach

1. Stage1: Understanding a logical boundary in a legacy monolithic system isn't direct. It has to be thoughtfully identified. Team resources and its structure also plays a role in the identification. It should be a self contained application with logical separation with its associated application(s).
2. Stage2: Once the capability is identified, disconnect all the active connection(s) that are point to point. Understand the contract of the capability with inter-dependencies.
3. Stage3: Enable a single point of entry to the capability, with all the protocols, including but not limited to, SFTP, MQ, HTTP and/or others. There shouldn't be any other way/option to interact with the capability. This enables, Anti Corruption Layer.
4. Stage4: Start building the capability with To-Be Architecture. Maintain a decision point to ensure that the traffic can be directed towards the new or old implementation. This ensures that both implementations can process the data, and produce the same results. Last, but not the least in this step, it has to be ensured that the data stored or presented is reconciled towards the end. For Mainframe to reconcile the data at run time, a technique of CDC (Change Data Capture) is leveraged. CDC Details are listed in the section below.
5. State5: Once running both capabilities in parallel for some time (no less than 6 months), move all the active traffic to the new capability. Do not sunset the legacy capability just until all the scenarios are actively available in the new capability and no issues are registered.

By following the above step:

- Incrementally modernize the system by replacing applications and services in a phased approach.
- The new system slowly grows until the old system is "strangled" and can be removed.

Benefits

- Provides a way to reduce risk when performing a transformation
- Keeps old services in play while refactoring to updated versions
- Adds uniquely new services while refactoring older services
- Ensures incremental success throughout the lifecycle of the program
- Last but not the least, have one single System of Record(SoR), though there can be multiple Authoritative Data Sources (ADS)

Considerations

- Additional complexity to ensure synchronization of systems during interim phases
- A high degree of Subject Matter Expert (SME) knowledge and involvement will be required. There are tools available such as AWS transform and CAST Imaging, that can enable faster dismantling of the existing application within a

system. Details are listed below in the section.
- Enable ubiquitous data modelling when interacting amongst the application(s). Details are listed in the section below.
- Dual test harnesses to validate legacy and new services during interim steps
- Dependency on mainframe will continue to exist during interim phases
- Ensure all scenarios are tested in both the systems, therefore Test Data Management (TDM), and Test Harnesses are crucial for the success. Details are listed in the section below.

5.3 Ensure that the current state is fully captured with CAST Imaging or AWS Transform.

CAST Imaging

1. To expedite the discovery process, and draw boundaries around applications that are part of a monolith, tools like CAST Imaging can be leveraged.
2. Tools like Cast Imaging (Figure 4) enables Refactor current platform to a target platform that will be "Cloud Enabled" and Containerized. Predominantly target tech stack to be Java-based frameworks, fit for purpose datastores, along with other capabilities. [6]
3. Cast Imaging provides help in all the dimensions listed below:

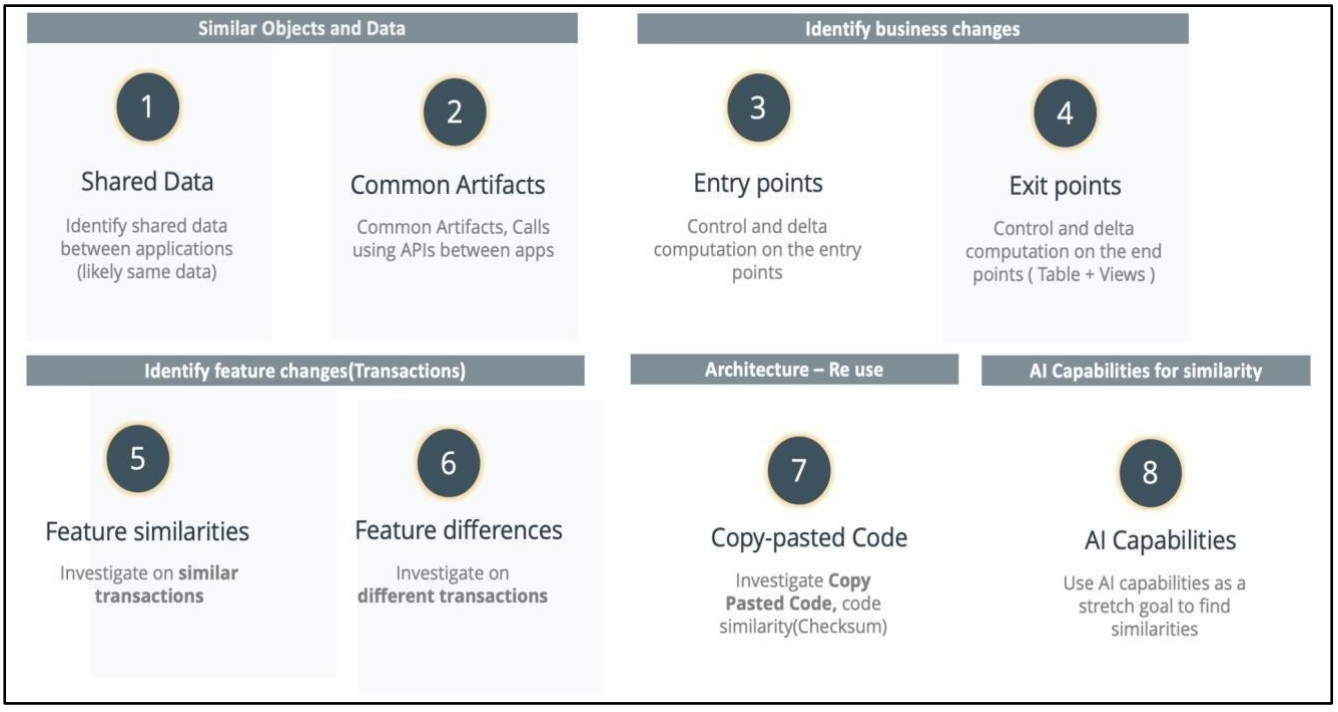


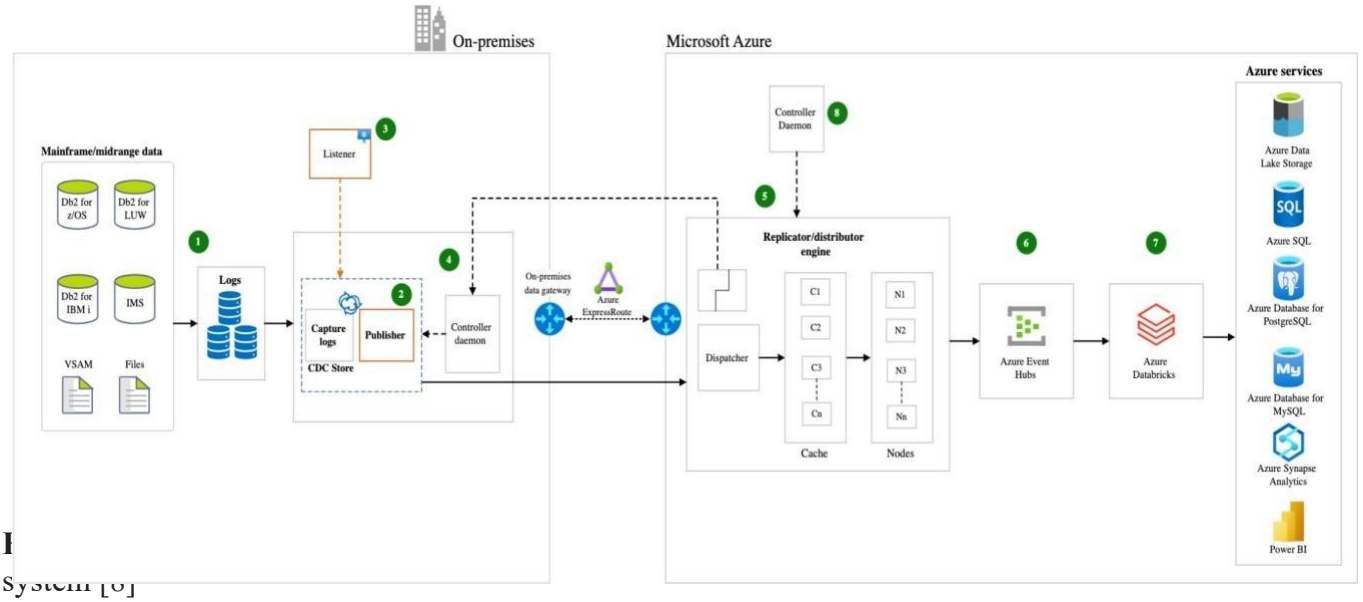
Figure 4. Analysis via CAST Imaging

AWS Transform

AWS Transform enables holistic overview of a program. Creates the flow diagram and Business Rules Document. This enables the faster discovery process [7].

5.4 Extract Data in Real time from Mainframe using Change Data Capture

To perform reconciliation from Mainframe to the distributed system. A concept of Change Data Capture (CDC) is being leveraged (Figure 5). There are various tools available in the market to perform this CDC, such as IBM CDC, Precisely, etc.
CDC tools, when configured against a VSAM file or DB2 table, listen to the audit logs, and as soon as it encounters a change in the enable replication parameters, it pushes the data in the target agent. Various target agents are supported by tools, all the way from Kafka to relational DB [8].



System [8]

5.5 For the lower environment, test data management

To ensure that we're testing system(s) in parallel, it becomes important that we get the data from production, to ensure all corner scenarios are tested and executed by the new and existing application code for comparison.
The synthetic data can still have some scenarios that might not be captured; therefore, downloading obfuscated data from production into a lower environment ensures lower error rates in PROD.
The diagram below (Figure 6) depicts a view of creating test data that is completely obfuscated.
Step involved:

1. Define the sensitive data fields that have to be obfuscated.
2. Extract the data from the existing prod system into CSV file(s).
3. Export the files in an object store such as AWS S3.
4. Use an obfuscation engine, such as Delphix [9], to obfuscate the data. Ensure the data integrity is maintained during the obfuscation process.
5. After obfuscating the data, use AI to compare the original data to the obfuscated data, ensure there is no PII or sensitive data exported in obfuscation.
6. If sensitive data is found, it is again sent back to be processed via an obfuscation engine, until cleansed data is generated.
7. After data obfuscation is completed, RAW data is then converted into the target state data model, using an AI conversion engine, that can be then executed by the new capability.
8. The raw obfuscated data will be leveraged by the legacy capability implementation that has to be strangled.

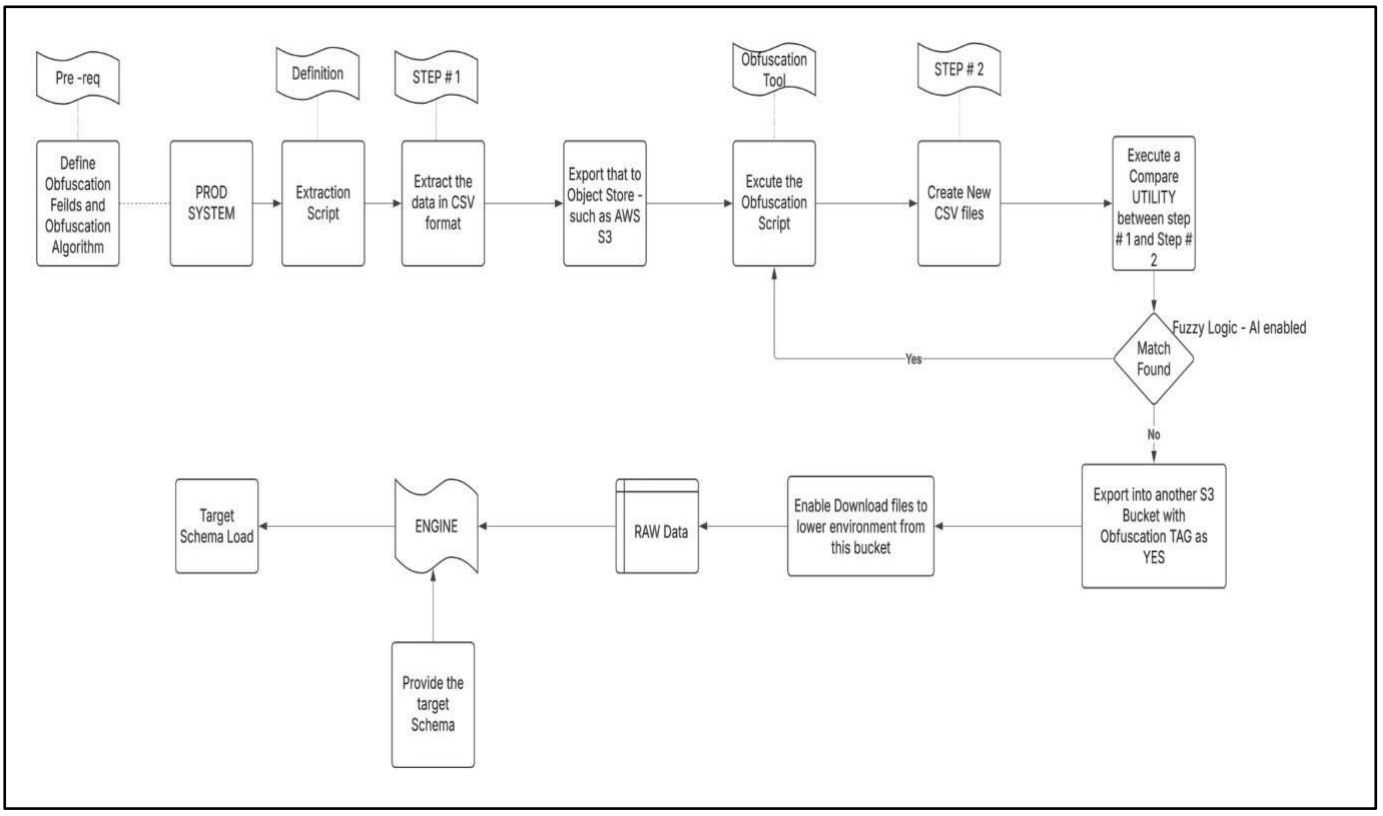


Figure 6. Steps for Test Data Management

6. Conclusion

The modernization of legacy mainframe systems is not a choice anymore for the enterprises aiming for agility, and digital advancement. The Strangler Pattern 2.0 offers a practical approach that is incremental in nature to transform a complex application that has embedded business rules. It is a balancing act of risk acceptance, cost management, and business continuity. By allowing systems to coexist, legacy and modern components, this pattern allows enterprises to decompose complex legacy architectures, induct latest technologies, and migrate functionality without disrupting critical business functions. Managed by strong governance, automation, and capability-driven design, the Strangler Pattern 2.0 provides framework for reducing risks for failure, enabling long-term value, and in parallel providing short term wins. As organizations continue to evolve, adopting this pattern can serve as a strategic enabler for sustainable transformation at scale.

References:

1. https://www.tcs.com/who-we-are/newsroom/press-release/global-cxos-mainframe-legacy-modernization-top-business-priority-tcs-survey
2. https://www.paramatrix.com/blog-list/the-hidden-costs-of-legacy-systems-why-modernization-is-crucial
3. https://www.microfocus.com/media/brief/how-mainframe-modernization-begins-with-application-modernization-research-brief.pdf
4. https://www.gartner.com/smarterwithgartner/7-options-to-modernize-legacy-systems
5. https://guides.visual-paradigm.com/business-value-assessment-technique-in-the-togaf-framework/
6. https://www.castsoftware.com/imaging
7. https://aws.amazon.com/transform/mainframe/
8. https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mainframe/mainframe-replication-precisely-connect
9. https://www.perforce.com/p/data-masking-contact-clone?utm_source=googleadwords&utm_medium=cpc&utm_campaign=Delphix-DataMaskingDemo-NORAM&utm_adgroup=DataMaskingDemo-NORAM-search&utm_term=data%20masking%20solutions&gad_source=1&gad_campaignid=21606582618&gbraid=0AAAADyv2jwthTJITnWBwsOlVKDxxR8co&gclid=CjwKCAjw7fzDBhA7EiwAOqJkh5SSGcr91Zr_sQKG_vpg6at6AKTOFAHsxoro3ObyLHKW1_kM7pGY6RoC7psQAvD_BwE
10. https://www.in-com.com/blog/mainframe-modernization-for-business/#:~:text=Legacy%20mainframe%20systems%2C%20while%20reliable,changes%20in%20demand%20more%20dynamically.