

I. Section 4 – Understanding and Approach to M&E Services

4.2 M&E Understanding and Approach to Application Evolution (4A)

RFP # 5.3.3.2 (RFP Table # 41)

We have worked collaboratively with the Consortium and CalSAWS contractors—as One Team—to achieve substantial outcomes. As one of the first steps of the CalSAWS migration, together we migrated the application workloads from an on-premises data center to cloud-based hosting, creating greater opportunities for evolution of the CalSAWS application. The Consortium is a leader in technology evolution, from your early investment in web technology for the C-IV System to your adoption of cloud technologies. We are proud to have supported your journey, and we are the right partner to continue supporting you. The following guiding principles have shaped our approach to evolving the CalSAWS application using a cloud native microservices architecture:



Your Success Accelerated

- **Unmatched knowledge** of your business operations, your system, and the supporting infrastructure
- **Low Risk** with an advanced, proven, decoupling approach to modernization and leading technical expertise from AWS and our Accenture AWS Business Group
- **Faster value realization** with unmatched delivery timeline
- **Strengthens Security** through an integrated DevSecOps

Advance and evolve CalSAWS Architecture: By introducing a modular architecture and using an automated DevSecOps model, the CalSAWS system will be **easier to maintain** with less impact across system features.

Strengthen Operational Security: By continually enhancing and improving world-class security processes and standards, we help **protect our customers' information** from the ever-changing threat landscape and enable counties to continue providing top tier service.

Lower cost of ownership: By moving to a **cloud-native** serverless environment, the Consortium will **reduce its software licensing and environment maintenance spend**, freeing resources to be redirected to business and legislative changes that keep counties running efficiently.

Increased User Efficiency: A future-proof system means that the Consortium and development teams can **quickly respond** to changing policies and rapidly evolving user needs due to unexpected issues such as a pandemic or wildfires.

Table 4-1 shows the overarching themes—Acceleration Essentials—of our Application Evolution approach for CalSAWS.

Table 4-1. The Features (What We Bring) and the Benefits (What You Get) of our application evolution approach will enable a future-proof system.

4.2.1 Breaking Down the Application

Item # ME-UA4

Describe your strategy and approach to breaking down the large CalSAWS application into feature modules, prioritizing, and refactoring the application to evolve the application architecture.
Describe how this strategy will be integrated with the DB Migration effort including decoupling the CalSAWS application from the database structures into feature modules.
Describe how this strategy will address security considerations, reduce costs, and improve optimization, scalability and flexibility.

We have carefully read the application/architecture evolution requirements in RFP Attachments A2 and B2, the SLA requirements, as well your vision for the next chapter of CalSAWS. We understand that the Consortium seeks to evolve the CalSAWS application and architecture from a monolithic to a modern, scalable, and dynamic cloud-native application architecture. The Consortium expects an approach that maintains the CalSAWS application and architecture throughout the evolution, using automation, artificial intelligence, and machine learning to automate manual tasks to increase accuracy, reduce costs, and improve performance and the user experience. Our approach contains these elements, and additionally furthers the event-driven architecture for operational independence to exceed your requirements.

We propose a solution approach for a rapid, high-quality, and low-risk evolution to a microservices-based architecture running smaller applications on independent databases. The microservices will be based on logical business domains. For example, where Journal functionality is part of the monolithic application today, it will be a separate microservice in our proposed architecture. The new architecture will be modular, more secure, and easy to maintain and will reduce the overall cost of ownership. Our proposed architecture changes will lead to faster developed and deployed system changes. Additionally, they will improve all processes and events, such as case management, batch jobs, task management, analytics and reporting, and security. During and after the architecture migration, staff from the 58 counties will continue using a system that enables them to serve their customers in a timely and efficient manner.

In determining our overall approach to application evolution and database migration on AWS, we arrived at **three strategic decisions**:



CLS IME 22.0213

1. Will we perform a Big Bang cutover, or will we incrementally modernize the application? As depicted in Figure 4-1, incremental delivery provides early benefits aligned with the CalSAWS vision, reduces risk and cost.

Big-bang vs. Incremental Approach

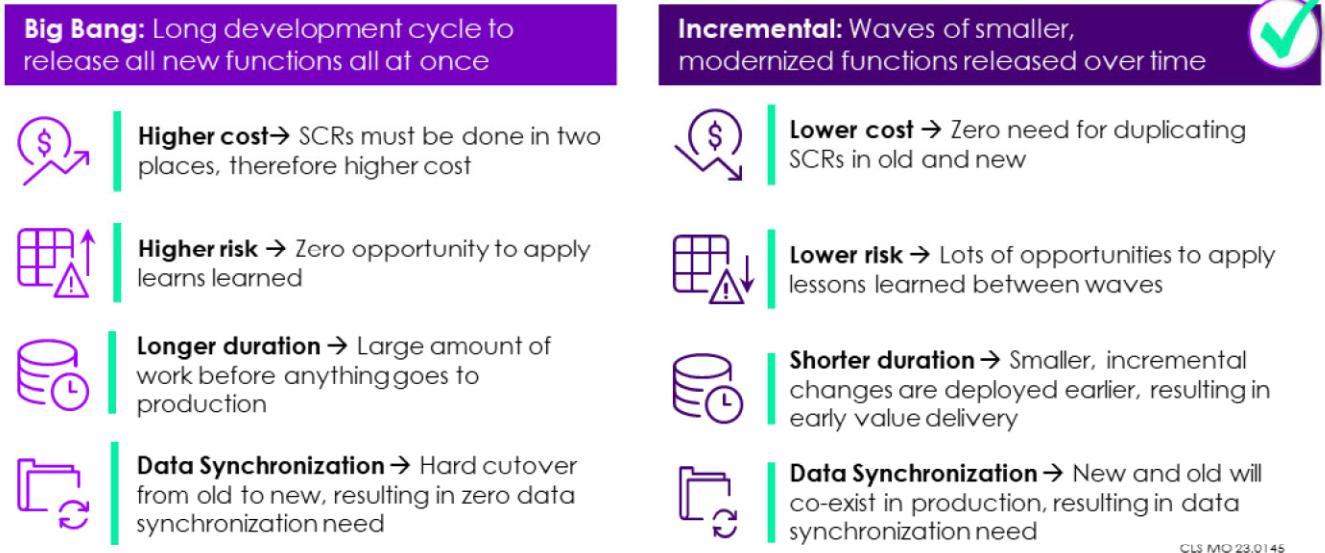


Figure 4-1. Incremental delivery provides benefits early in the evolution process, reduces risk, and lowers cost.

2. Will we first complete a database migration off Oracle to a cloud native database(s), and then return to modernize the application and middleware—or will we complete this simultaneously? As shown in Figure 4-2, an integrated versus siloed approach of modernizing the application and based on the benefits an integrated approach brings early value and lowers costs and risk.

Silo'ed vs. Integrated Approach

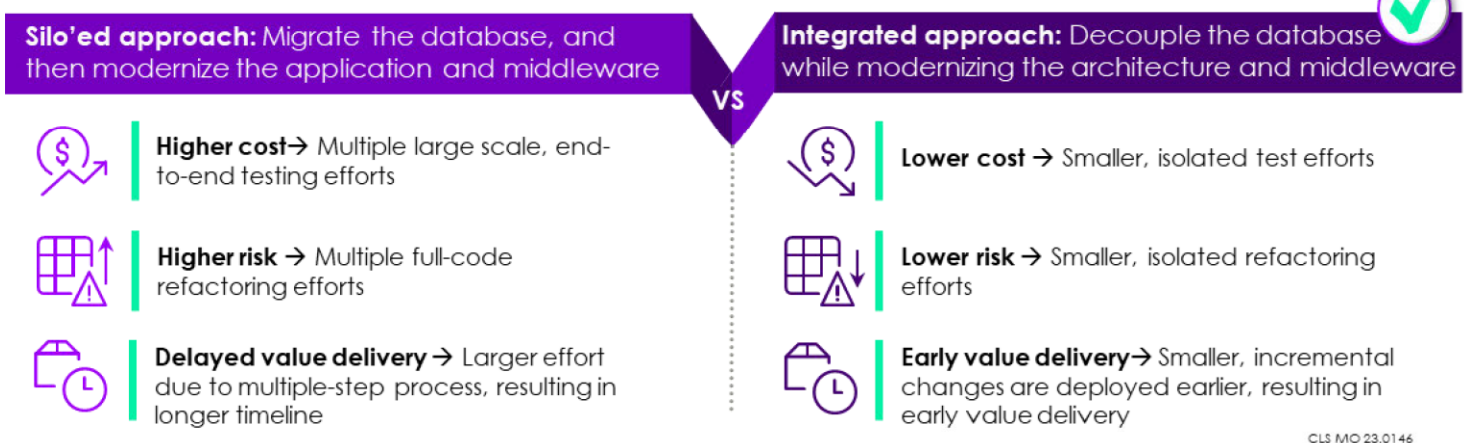


Figure 4-2. Integrated Modernization and Database Migration.

3. Will we decompose the application based purely on technical gains, or will we base it on what will provide the maximum business gains? Figure 4-3 shows a business led approach provides immediate business value and minimizes the maintenance cost.

Technical vs. Business Approach

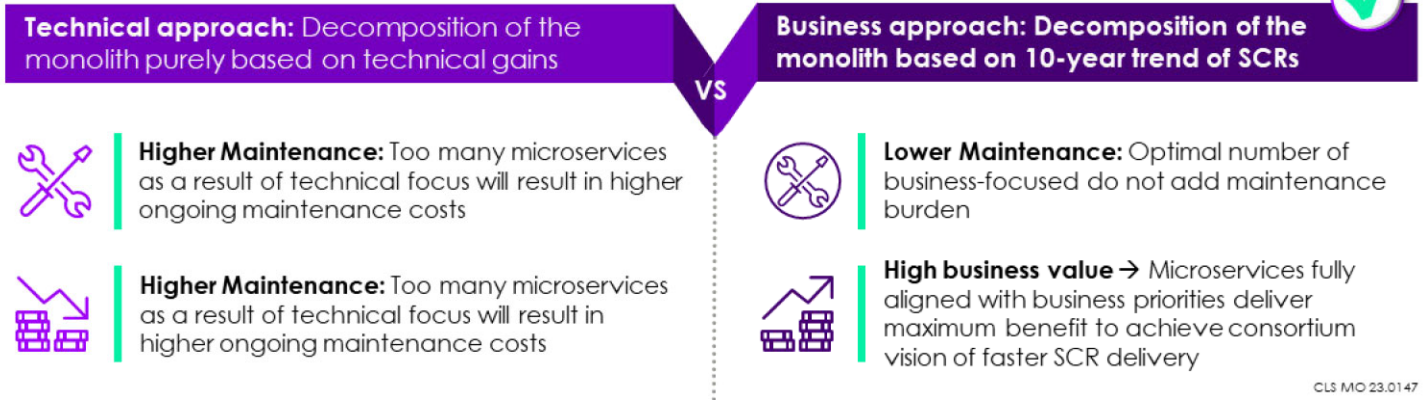


Figure 4-3. Business Led Evolution.

The elements of the application evolution are 100% AWS aligned. AWS services are in use today at CalSAWS and our approach continues to maximize the investments the Consortium has made and plans for in the future. The modernized application architecture is based on the overall strength the AWS offerings bring to bear through the continual innovation at AWS, Accenture AWS Capabilities, and our partnership with AWS. Figure 4-4 provides a summary of the technologies that will be used for modernization.

100% Alignment with existing AWS Architecture

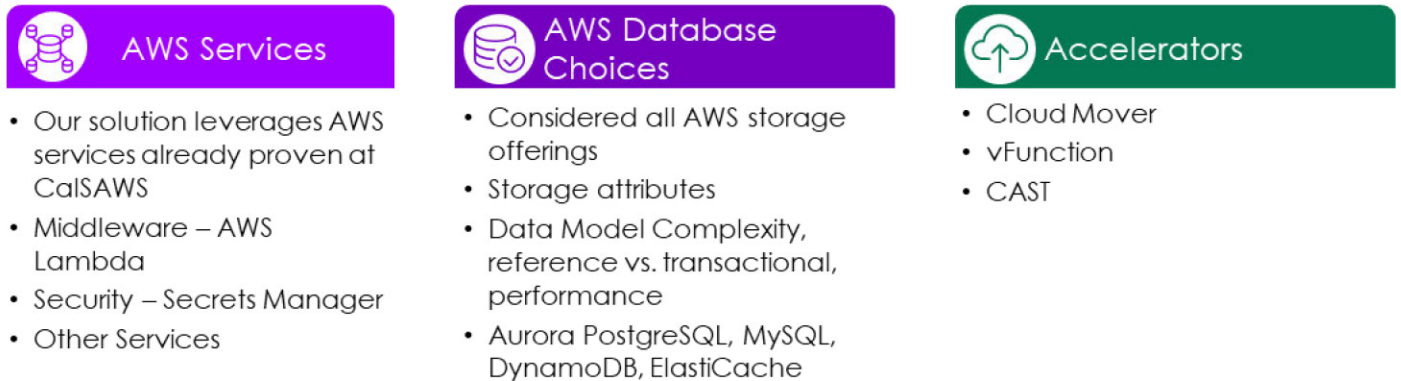


Figure 4-4. Our Application Evolution approach is 100% AWS aligned.

These strategic decisions informed and guided our approach to address and exceed your requirements. We will describe each of three major phases to application modernization and database migration, which will include how the application will be broken down into feature modules, our approach to decoupled databases and database migration, and how the application will be refactored into an optimal future state application architecture that is 100% aligned with AWS. We will also provide our proposed implementation timeline and explain how this approach will succeed in the multi-vendor environment and the benefits to the approach including addressing security considerations, reducing costs, optimization and creating maximum scalability and flexibility.

We have 10+ years of experience decoupling systems and understand how to build the new while running the “old” working with clients like HealthCare.gov.

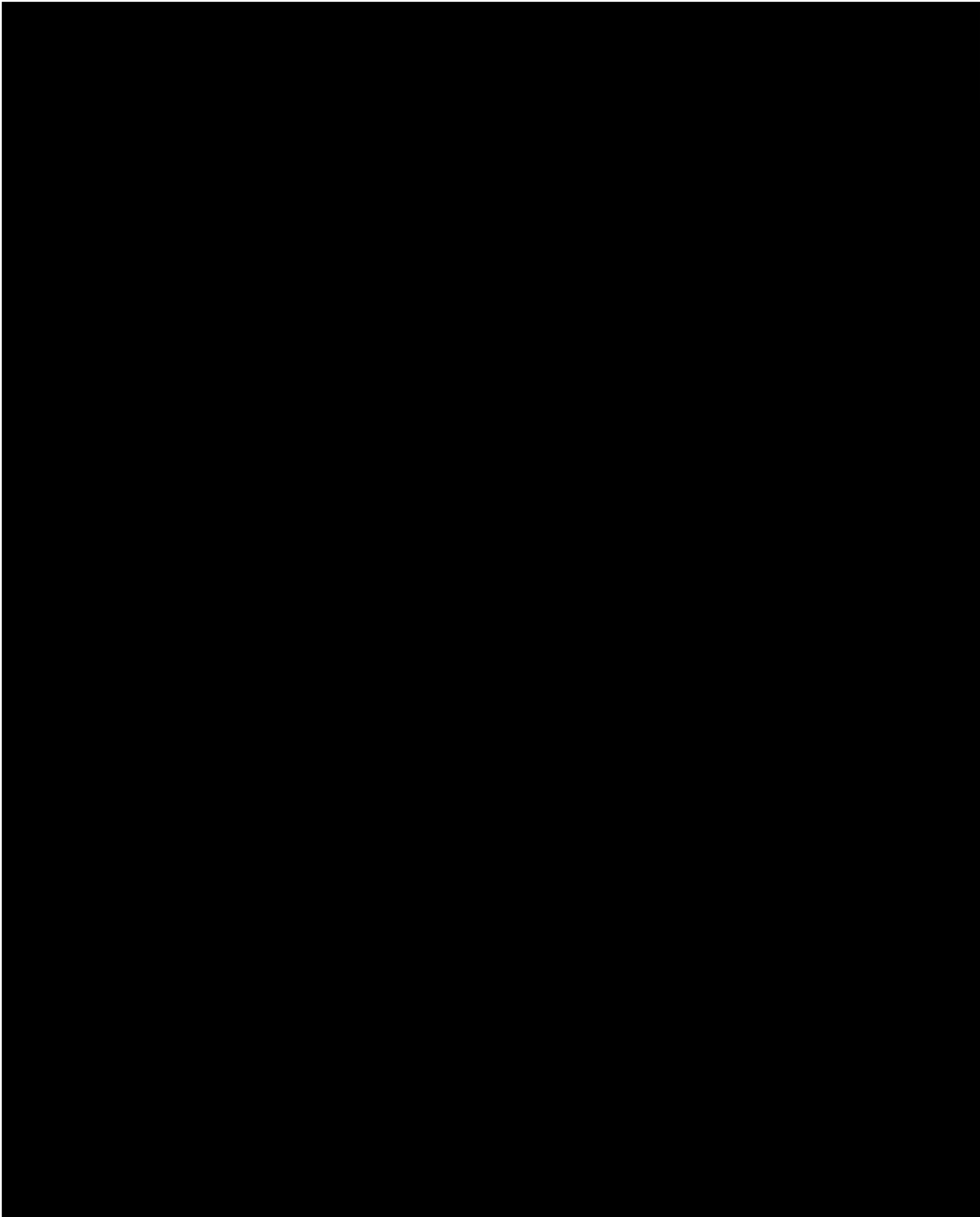


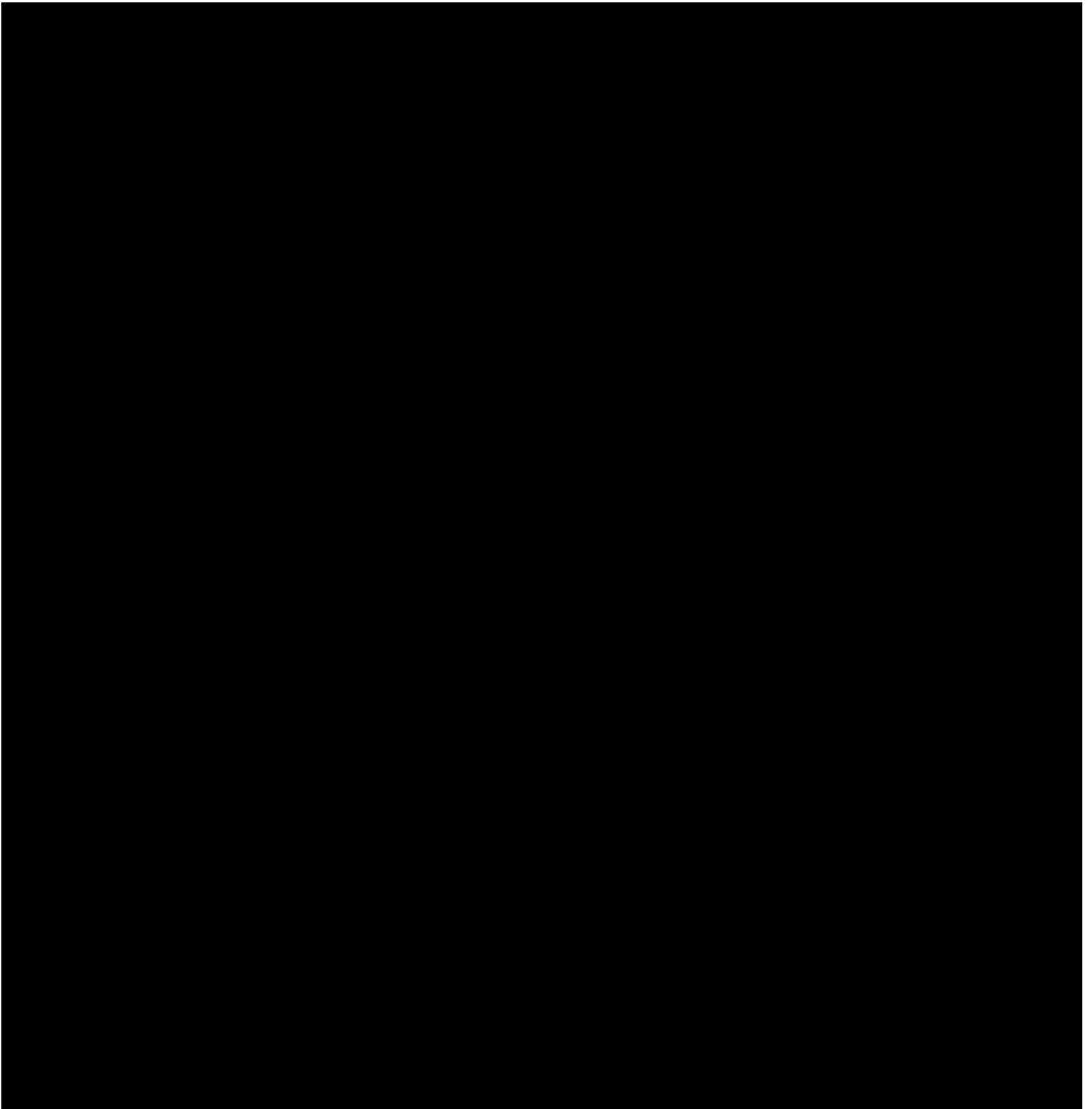
What Our Clients Say...

Accenture brought the best people to build, migrate the legacy data and support the successful implementation of the LA County Leader Replacement System (LRS). The implementation completed on time and on budget. LRS became the core system leading to the migration of the statewide California automated welfare system.

— Hayward Gee,
Former LRS Project Director

2 CLS IME 22.0233





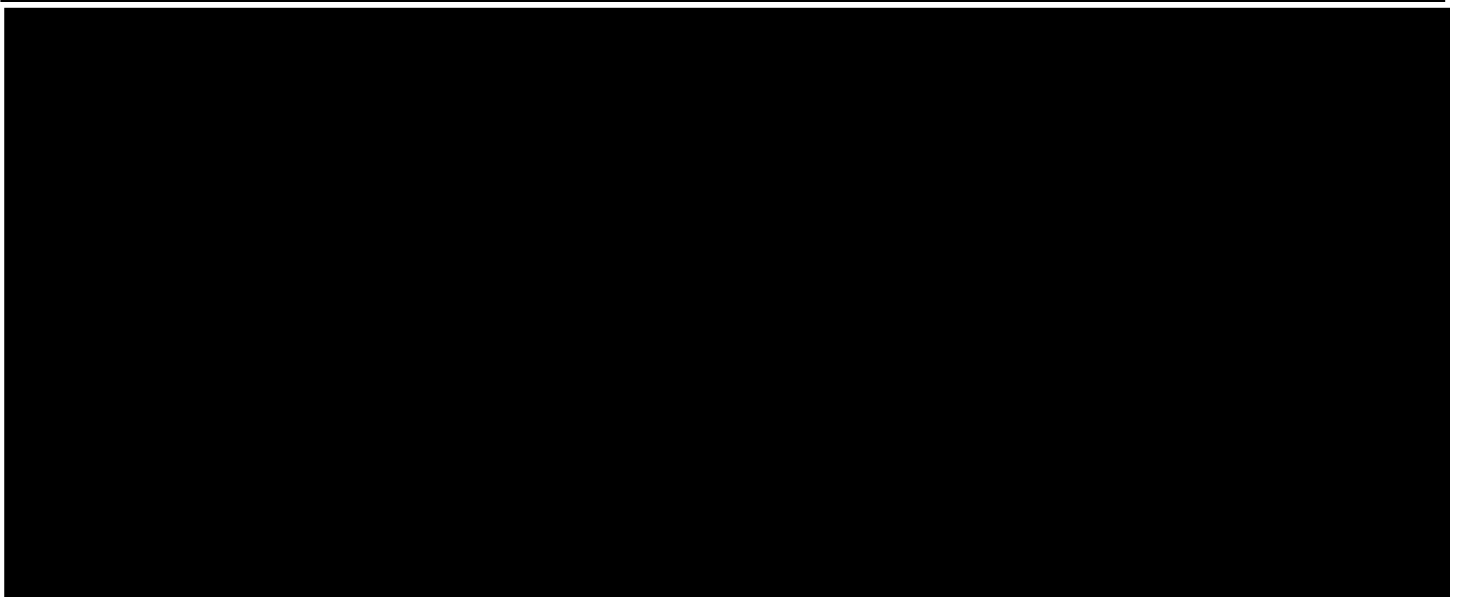
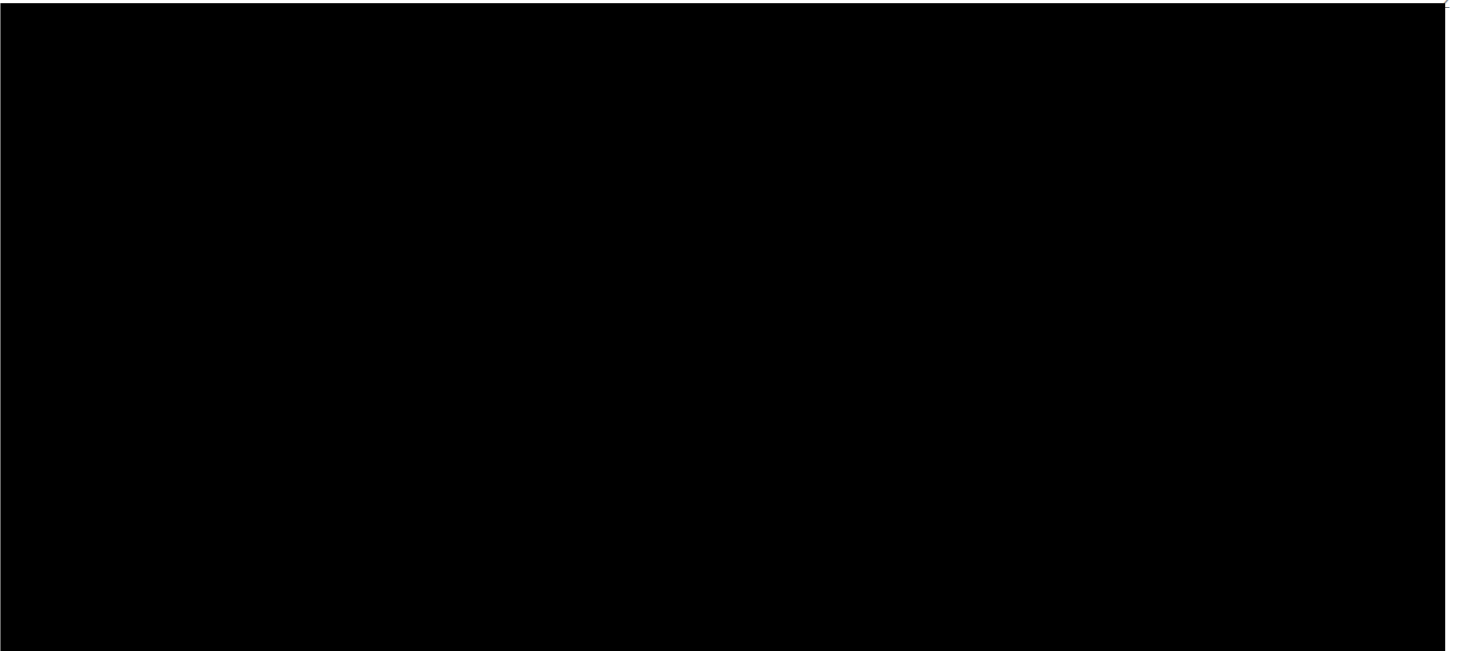


Table 4-2. We identify your business domains/functions and our approach to breaking them down.


Analysis Example – Identifying Task Management as a Microservice

As we begin the modernization approach, multiple analyses are performed on the CalSAWS application. The top-down analysis will review the business domains within CalSAWS to identify logical groupings of functionality. The vFunction tool will be used to perform a bottom-up technical analysis of the code. As an outcome of the of the analysis, our example identifies three unique business domains to be decoupled from the main application into microservices as identified in Figure 4-4. The business domains include:

Data Collection • Eligibility Determination and Benefit Calculation (EDBC) • Task Management

Using data provided from the vFunction tool which includes a risk and dependency evaluation, we decided to prioritize Task Management in our example as the initial microservices to create. This is because task management has limited downstream functional dependencies and limited risk based upon the analysis. The effort to decompose the task management functionality into a microservice is aligned to the Task Management Product Oriented Delivery (POD) team.



 We will start with delivering a decoupled and modernized Case Management business domain and associated correspondence changes. This will serve as the **proof case and establishment of the interim architecture to roll out the rest of the features to reach evolution completion and legacy deprecation.**

Design and Build for Coexistence of Legacy and New Applications

Accenture will design and build an interim architecture, supported by the Mercury services platform, that will enable the Decoupling approach—an iterative approach that simultaneously supports old and new functionality until what remains is the defined future state architecture. The Decoupling approach combined with the feature toggle enables the legacy system to coexist with the new

services while confirming new features have minimal impact on current functionality. First, we will use vFunction, the same tool used for our bottoms up analysis, as a Monolith Decomposition Tool (MDT) to decouple and create the microservices. Second, we will add an abstraction layer, enabling us to use either the legacy system or the new microservices. Next, we will configure a toggling feature, enabling us to turn specific functionality on and off so the legacy and new system can coexist, business operations are never disrupted, and priority SCRs can progress. Lastly, we will configure the feature toggle to block legacy routing and direct all actions to new microservices only. Figure 4-8 illustrates how this approach will work for a legacy function/method.

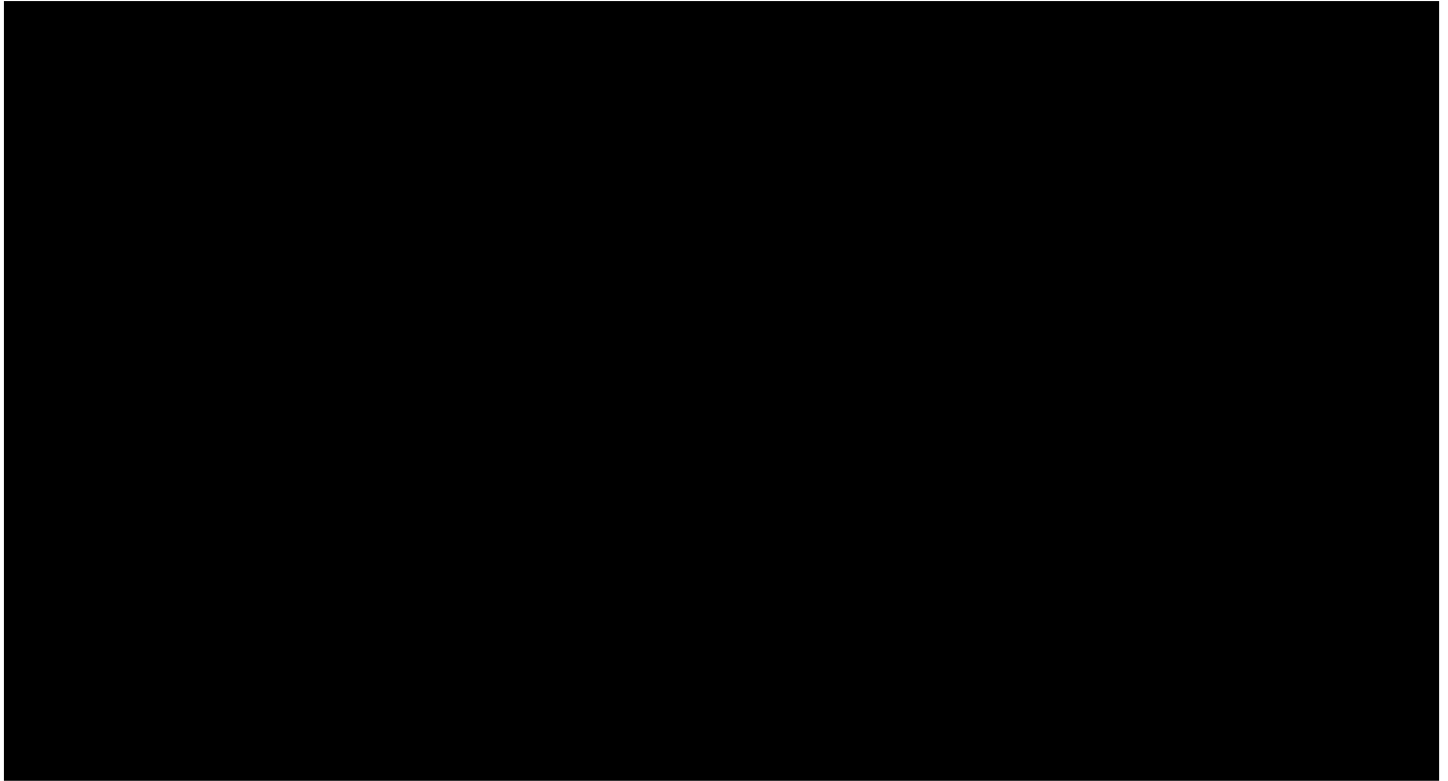
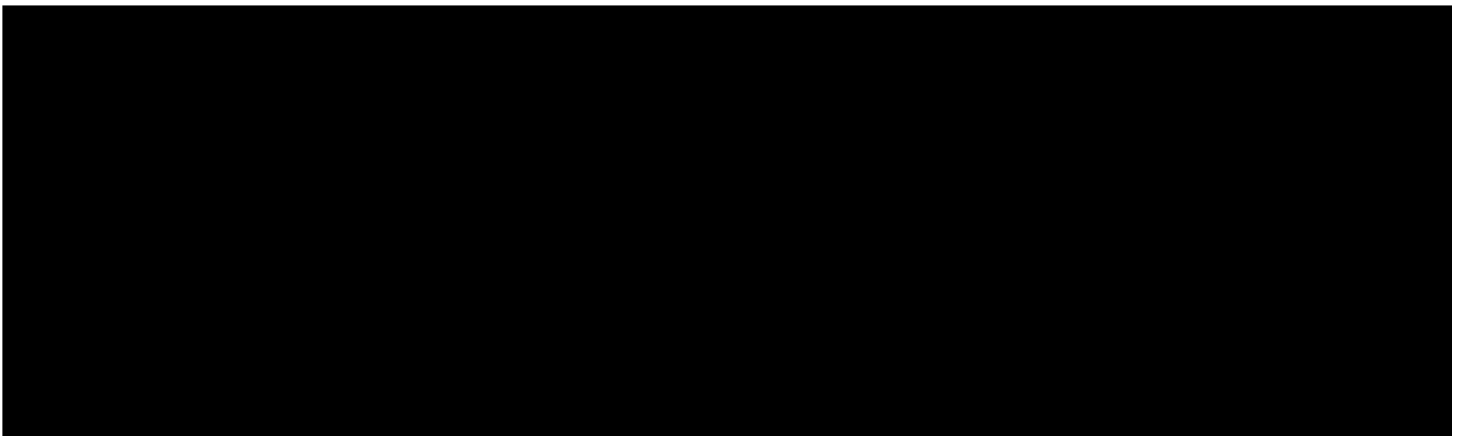


Figure 4-8.



This approach allows for iteration and minimizes delivery risk and will be completed for each feature in the legacy application.

Design and Build for Coexistence of Platforms



The microservices use AWS

cloud-native services in a serverless architecture to accommodate different patterns for the target applications. We will use the appropriate cost effective and optimized AWS service (e.g., Lambda) based on need. [REDACTED]

[REDACTED] the CalSAWS system will benefit from longevity, scalability, performance optimization, and cost efficiencies.

With our extensive knowledge of CalSAWS, we have initiated design and solutioning of the new architecture to qualify and validate our response, and to provide the Consortium with a thoroughly analyzed approach and solution—enabling us to hit the ground running, reducing overall time to achieve the target end state. Figure 4-9 shows an illustrative to-be CalSAWS architecture built and based on AWS Services.

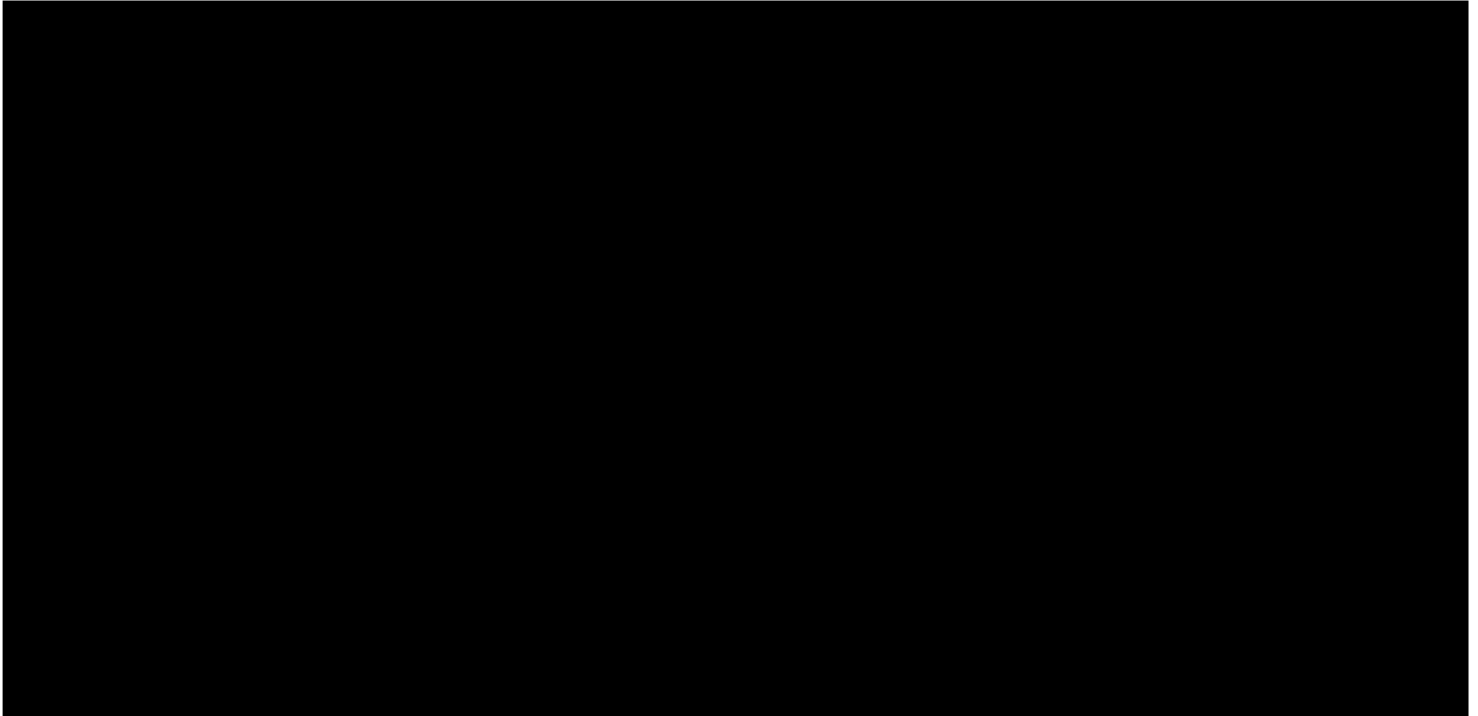
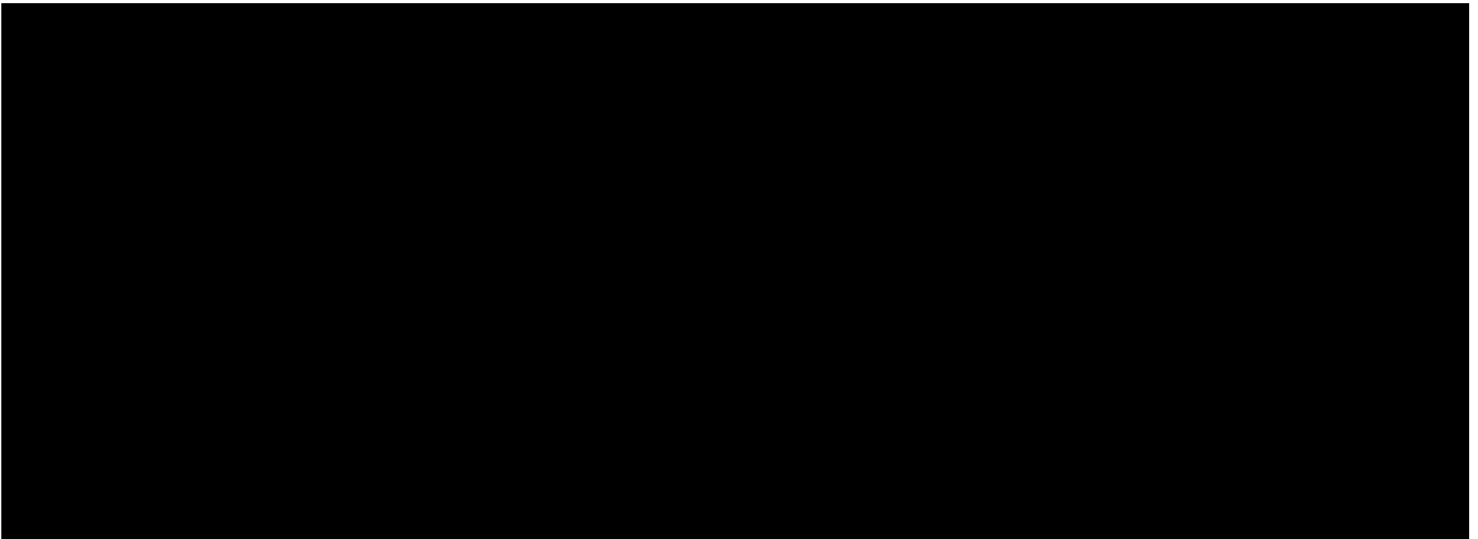


Figure 4-9. The [REDACTED] is centered around [REDACTED]
[REDACTED]

Creating the Modern Application



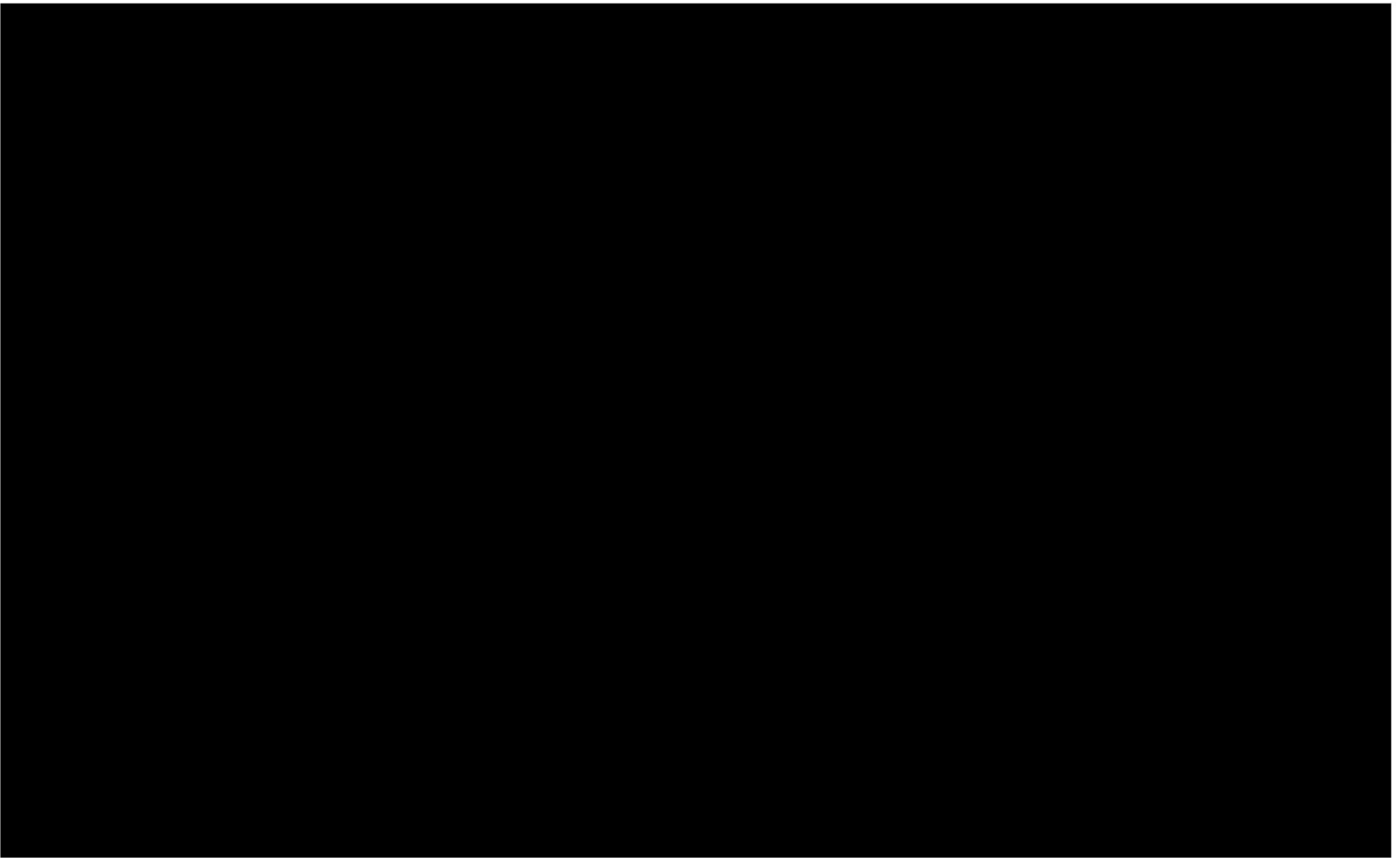


Figure 4-10.

Decomposition Example – Task Management

As illustrated in Figure 4-10, task management is part of the monolithic application in the current architecture. All pages and corresponding functionality are part of the large CalSAWS codebase. Additionally, the database tables that store the task management data are part of the monolithic CalSAWS database. This creates dependencies between this particular business domain and all other functionality within CalSAWS. In the current state, if an unrelated function causes an error making CalSAWS inaccessible, task management would be impacted even though it is an unrelated feature. In the future state, by decoupling task management into its own microservice, we will minimize dependencies between it and other parts of the application resulting in a more stable, performant, and cost-effective application. During the analysis phase, all features related to task management to be included as part of the microservice were identified. This is known as the bounded context. This includes all task management pages and underlying functionality. Additionally, the underlying database tables for task management will be decoupled from the legacy database and included in their own isolated database specifically for the task management microservice. Any portion of CalSAWS that interacts with task management will use an Application Programming Interface (API).

CLS MO 23.0141c



Traceability
and
observability

As noted previously, we will apply the SCR process to prioritize and implement application evolution changes. We will prioritize the SCRs created in Phase 1, for design and build in Phase 2, with the other SCRs within the same business area. This will promote flexibility—so project teams and stakeholders can effectively respond to priority SCRs during application evolution and focus time on the SCRs that matter the most. Additionally, following the SCR process to implement application evolution changes will confirm that security is part of the process through the lifecycle of the change. As stated

in Section 4.3 System Change Requests, security principles will start from design and run through build and testing. This applies to all changes following the SCR process including application evolution. This means that application evolution changes will be designed with security in mind, developer code will be auto scanned prior to checking in the code, and automated security testing will be conducted as soon as the code is deployed to the test environment. Identified security vulnerabilities will be treated as system defects.

Using a **Hybrid-Agile testing approach**, we will test changes as they are iteratively made to the system. Our team's extensive experience with integrated eligibility allows us to quickly iterate on and enhance our automated regression test suite, understand how to execute and what to look for in performance tests, and keep security in mind in our DevSecOps strategy—by continuously verifying application vulnerabilities through the development lifecycle.

We will apply this test approach to all application evolution changes which will continuously occur with every sprint. This reduces the need for defect re-work and re-testing, greatly reducing the time it takes to implement the change and increasing stability to the application by catching issues earlier in the development lifecycle. We have further described our testing approach in Section 4.2.2.4 How Changes Will Be Tested. Our Hybrid-Agile testing approach includes Sprint testing, Manual and Automated regression testing of the legacy and microservices, Manual Microservices testing, Security testing, Performance testing and User Acceptance testing.

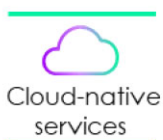
In addition to design and build for coexistence previously described, **Phase 2 will deliver capabilities for industrialization: continuous integration/continuous delivery (CI/CD) tools and deployment pipelines to support automated deployments; expansion of the existing automated regression testing framework to new functional areas such as batch and correspondence for a complete automated regression suite; and creation of a custom data synchronization process** so that production data between legacy and target application can remain in sync.

How do we coexist?

Using the Decoupling Approach, we will iteratively evolve the legacy system and move to the new environment over time to minimize the risk of disrupting business operations during the evolution journey. This allows for flexibility to modernize the application while still meeting business expectations on system operations and the ability to implement other priority SCRs.

CLS MO 23.00691

Decoupling and Database Migration



As we break down the application, we will simultaneously modernize the database to complement the application evolution. Each microservice will own its associated database and the data model will be designed for each microservice—and will not retrofit the legacy data model. As we decouple, the database migration will be integrated with app modernization and concurrently done. We will bring leading practices from Accenture's Cloud group, including learnings and advisory from Accenture Enkitec Group, a world-renowned specialized team in database platforms. [REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

Unlike many other HHS systems—including State-Based Exchanges that support largely MAGI based Medicaid/Medi-Cal programs where eligibility is straightforward and based on a minimal set of rules and data—the CalSAWS integrity eligibility system supports many public assistance programs which have much more complex eligibility rules, verification requirements, interfaces with external systems,

and supporting functions like benefit issuance and benefit recovery. [REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

Database Migration Example – Task Management

[REDACTED]

Implementation Steps of the Iterative Database Decoupling Strategy

Using the Decouple Feature, we have depicted the development, deployment, and clean up activities in Figure 4-11 which outlines how the strategy is integrated with the database migration approach and highlights the steps taken which correlate to the numbered activities using the Task Management Service as an example.

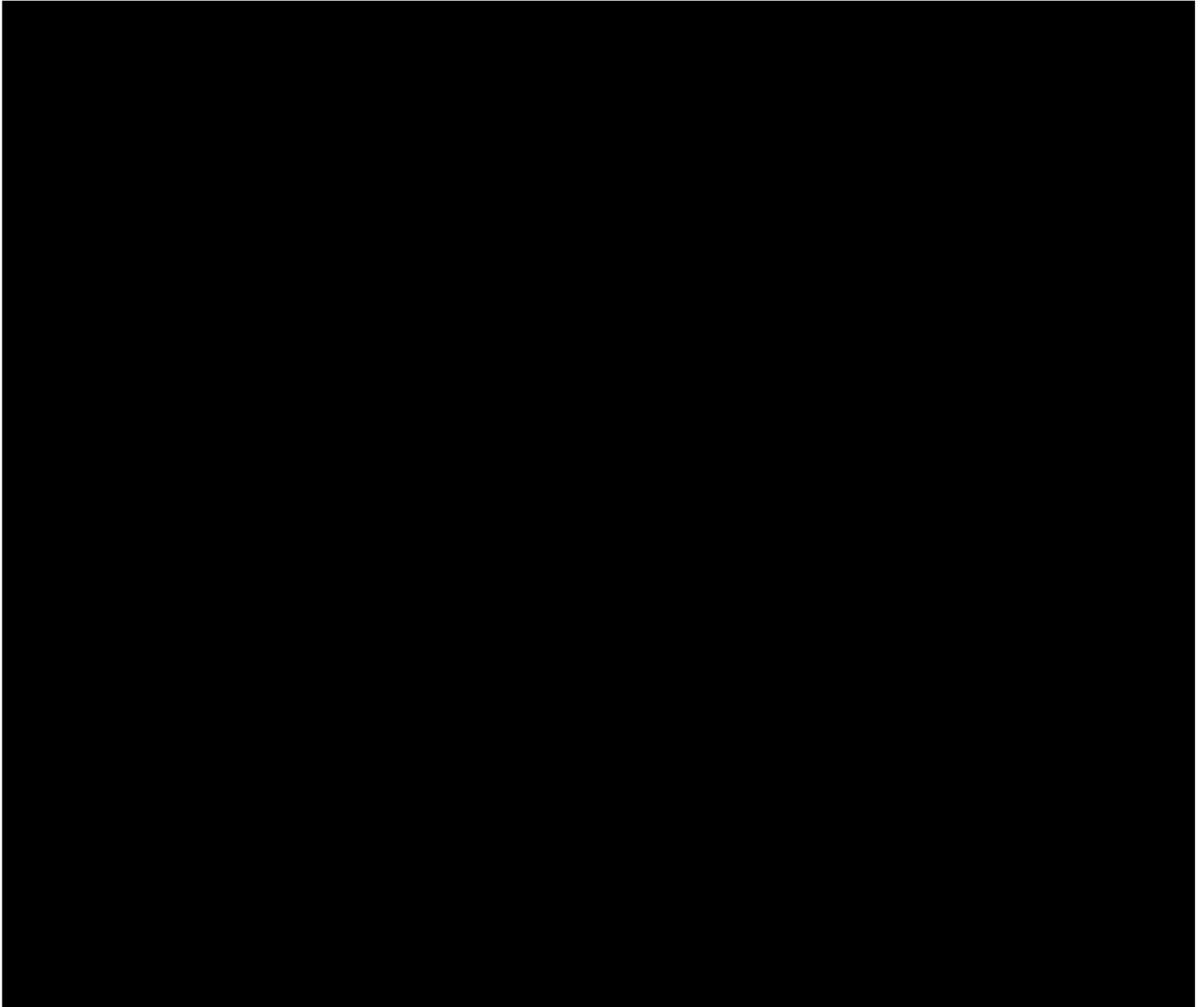


Figure 4-11. Our decoupling strategy implementation steps

We will scale the approach using continuous innovation and automated deployments to modernize the architecture, the business rules engine, and the database with security throughout the change process. This will enable us to scale the design patterns and microservice architecture for the remaining microservices to achieve the target state architecture. As the journey progresses and we evolve the application and database to target state, the corresponding legacy system components will be retired, as well as the use of the Mercury services platform.



Siloed versus Integrated, why not a lift and shift of the database? A phased approach minimizes delivery risk to business operations in comparison to a big bang approach as changes are completed in smaller and more controlled portions over time. This iterative approach aligns with the proposed hybrid-agile SCR SDLC, embeds security throughout the lifecycle of each change as described further in Phase 3 Industrialize, provides greater transparency to the Consortium, and fosters collaboration throughout the transformation journey. Lastly, it is critical to enable progress on other priority SCRs from the State and Consortium as application evolution is conducted. Working iteratively allows us to better plan, implement, and test both SCRs and application evolution changes simultaneously.

CLS MO 23.0069K

Our proposed approach considers leading practices and lessons learned from our previous engagements to define a clear path to evolve CalSAWS into one that is simpler and modular. By the end of the evolution journey, all business operations will be moved to the modern, scalable, and dynamic cloud-native application architecture. We will complete this in an automated and industrialized manner, aligning with the rules and principles, eliminate the possibility of manual errors during deployment, and improve operations and support using an agile workflow to quickly respond to changes. This allows CalSAWS stakeholders to monitor work at the transaction level, where tracing will proactively help detect anomalies and provide clarity of what may be causing them. All of this will provide additional insight on system optimization and performance.



Application Evolution Implementation Timeline

Did you know?

Accenture and CalSAWS have collaborated on a technical plan to pilot decomposing the Journal and Task Management functionality into microservices today. This Pilot begins in September 2023 and will demonstrate how microservices will seamlessly integrate into CalSAWS. Lessons learned will be applied to the effort to evolve the application in the future.



The initial months of this process also run in parallel with the SCR Hybrid-Agile SDLC implementation.

By the Industrialize phase, all application evolution changes will be executed through the new SCR process as defined in Section 4.3 System Change Requests. No other contractor can reasonably assert their ability to complete this application transformation within our proposed timeline. It would take over a year for another contractor to understand the relationships between business operations, data and business logic dependencies, processes, and monolithic code for the CalSAWS application.

Figure 4-12 illustrates our proposed implementation timeline and wave approach which we will discuss and finalize with the Consortium upon project start.

Database Decommissioning

Figure 4-12 shows the timing of

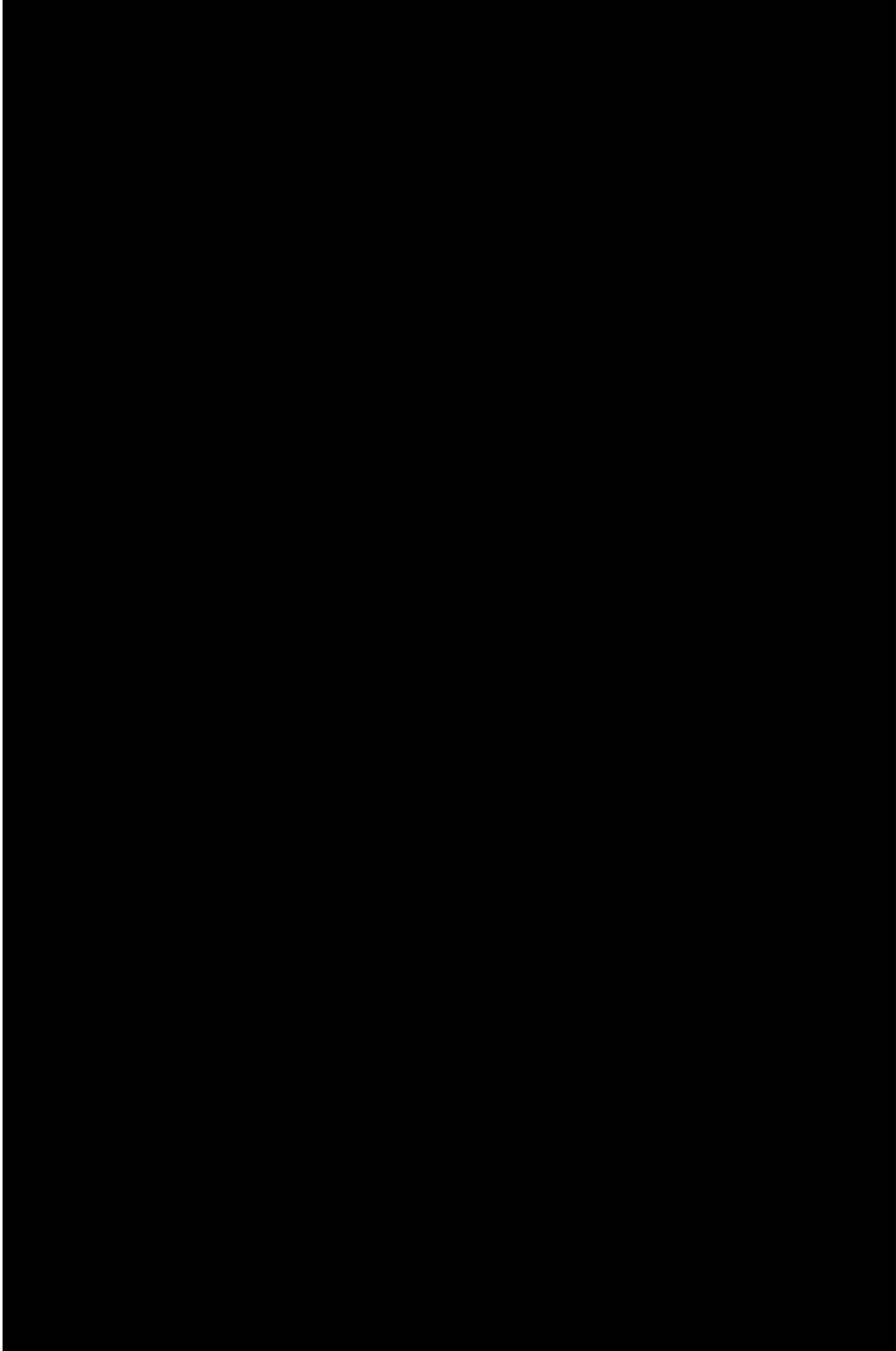


Figure 4-12. We propose

[Redacted text]

[Redacted text]

Delivering Digital Decoupling Results

For a global travel company, we tackled their mainframe modernization challenges using a digital decoupling approach and moved their legacy capabilities to a cloud native system. The client went from an inflexible and cumbersome system to a globally distributed reservation system on both their private and public cloud, providing better response times, the ability to innovate and adopt new features, and keep up with market demand. As a result of our design solution, this system is currently being extended to other public cloud providers.

CLS MO 23.0069n

Staffing: Our Application Evolution Team Structure

Given the complexity of development required to modernize each domain into the 92 microservices, we have structured the Application Evolution Team with dedicated specialized resources in different technical areas with the skills and experience necessary to help drive design and development decisions.

Architecture Team: Our Architecture team, comprised of dedicated Architects across different specialized areas and AWS Solution Architects, determines systems best practices and processes that will be leveraged throughout the modernization effort, drive architecture and design reviews, create the architecture blueprint, and assist in rules engine and end-to-end integration design for each of the application domain areas. The AWS Solution Architects help co-create our architecture and design to verify we are using the latest knowledge of the AWS platform roadmap. In addition, AWS is a strategically included professional service and is also available to provide guidance as an advisor as we modernize CalSAWS to microservices.

Microservices Developers: Our Application Evolution PODs will be staffed with dedicated Microservices Developers who range in specialization such as Batch Development, Integration Developer, Rules Engine Developer, Cloud Engineer, and Microservices Development. Our Microservices Developers will be focused on driving detailed design for the code refactoring as we upgrade and untangle the code. Our Microservices Developers will work collaboratively with the Database Modernization team members and analyze and provide input on the current database schema and design a new data model to align with the microservices, data mapping, and data migration activities.

Database Modernization Team: A significant part of our microservices approach is decomposing and decoupling the database tables. Database Modernization resources will be aligned to drive end to end data modernization from Oracle to the cloud native PaaS designed for microservices.

Performance Engineering Team: With our commitment to minimal disruption to the counties, we will dedicate Performance Engineers to confirm performance engineering best practices are utilized

throughout the Application Evolution SDLC. The Performance Engineers will also drive any needed performance testing based on the domain being modernized to microservices.

Integrating Our Full Stack Team with Application Evolution

Functional SMEs from our Full Stack Teams, such as Justin Dobbs for Task Management, and our Application Evolution Team members will work collaboratively in the development of the 92 microservices. Functional SMEs will be aware of the work in the pipeline for our legacy application and verify decisions are made to eliminate disruption to on-going maintenance and enhancements. Just like our Full Stack team structure and SCR Approach, we will be building integrated PODs to modernize each microservice domain using our Hybrid-Agile, DevSecOps approach. As an example, one such POD will be the Task Management POD. This POD is assembled with a Functional SME for Task Management, staffed, and supported by Application Evolution resources from Architecture, Database Modernization, and Performance Engineering Teams, and has dedicated Microservices Developers. This Task Management Application Evolution POD will be multi-skilled to handle all components of our proposed SDLC—from design to deployment to production.

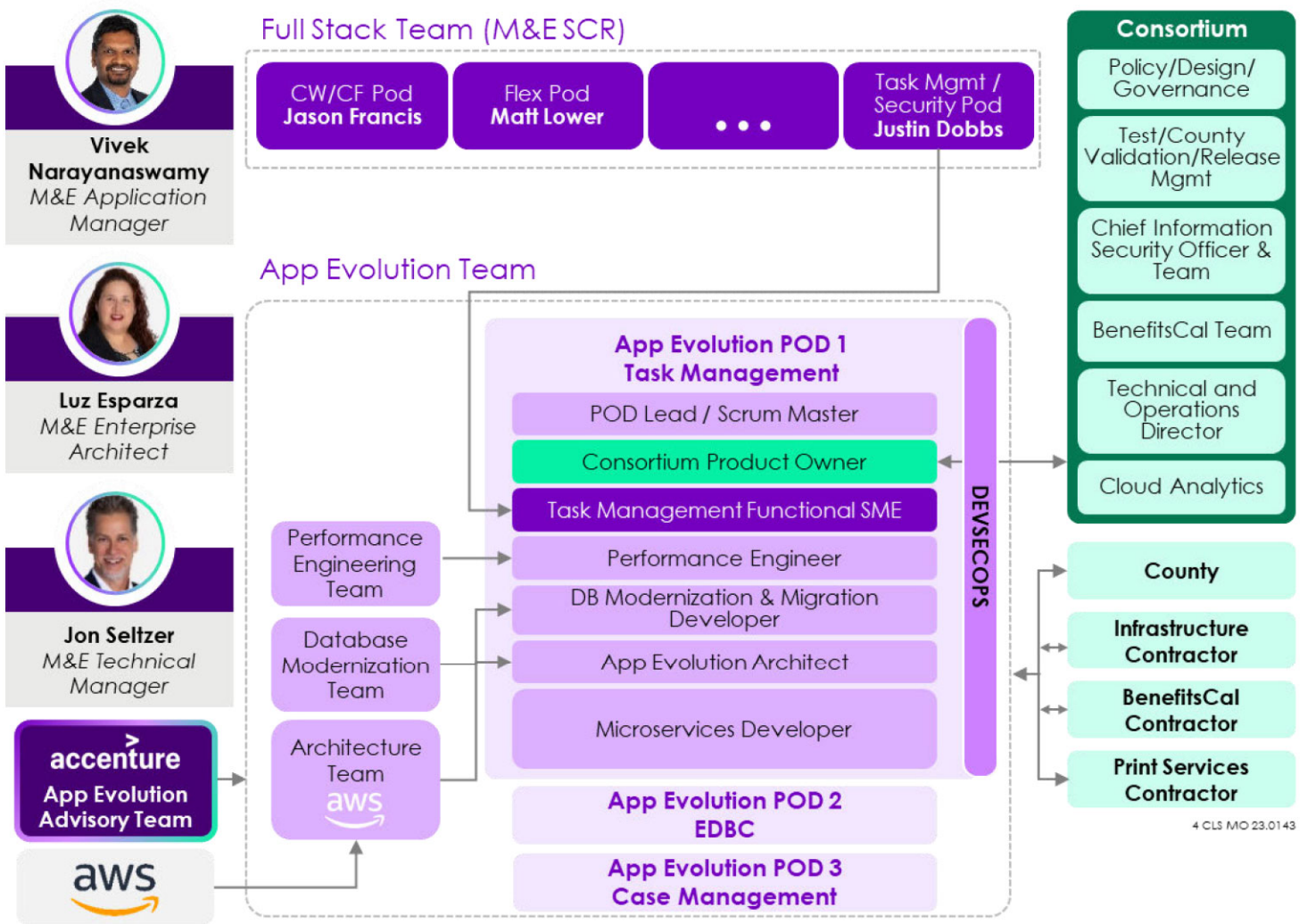
Accenture Application Evolution Advisory Team

In addition to our highly specialized dedicated resources, we will have advisors available to provide guidance in all aspects of the SDLC. This team of resources has successfully delivered on modernization efforts, microservices, and has AWS expertise coming from our Accenture AWS Business Group (AABG). This Advisory Team will help develop a strategic plan and bring in best practices to equip our already tenured and experienced resources with the right skills and knowledge to make decisions to successfully modernize the CalSAWS application.

Application Evolution Using a Hybrid-Agile Approach

An important aspect of our application evolution approach is using our proposed SDLC to deliver work. Each POD will have representation from either the Consortium and/or County and will be involved in the entire lifecycle. Given the uniqueness of each domain and the complexity of the work, we will involve the right Consortium Business Analyst and/or County stakeholder. Where necessary, we will include the Infrastructure Contractor, BenefitsCal Contractor, and the Print Services Contractor, depending upon the domain currently being modernized to microservices. Collaborating with the Consortium, County, and other contractors in our Hybrid-Agile SDLC helps garner support through microservices change requirements and design, accelerates implementation with iterative engagement, and increases collaboration and transparency to any identified issues.

A visual of our cross-team collaboration, support from our Accenture Application Evolution Advisory Team, Infrastructure Contractor, and Consortium and/or County stakeholders is depicted below in Figure 4-13.



4 CLS MO 23.0143

Figure 4-13 Our Accenture Application Evolution Team collaborates with the Consortium and Counties and is well supported by our partners and firm leadership for implementation success.

Collaboration with Counties and Other Contractors

We understand the relationship between the Consortium and the counties, and we know that **communication and transparency are critical for the success of the application evolution journey**. As we initiate this process, we will conduct a kickoff with the Consortium, other contractors, and counties. The kickoff will communicate the application evolution objectives, high-level plan, approach for delivering application evolution changes through the SCR process, roles and responsibilities, and expectations from an ongoing communication plan through the evolution journey. As we develop a cohesive plan for implementation, we invite all impacted stakeholders to provide feedback before the plan is finalized and address and resolve any concerns that are communicated to ensure we are all in alignment. We will begin executing the aligned communication plan with the Consortium, other contractors, and counties. We will additionally review progress, accomplishments, and risks on a weekly basis during status meetings, and as needed in other governance meetings. We will engage the Delivery Integration Office (DIO) early in the process and invite input, oversight, and feedback to optimize the flow of communications amongst the teams.

Modeling after the SCR approach detailed in Section 4.3 System Change Requests, Accenture will work with the Consortium during each application evolution SCR's solution planning stage to identify key stakeholders within the project that will participate in all phases of the SCR, from solution planning through UAT and deployment. Our team will continue to work transparently with the counties and the

multi-contractor teams, informing them of progress and dependencies in a timely manner. To minimize the impact to dependent applications, we will use existing API definitions to verify that applications managed by other contractors will continue to consume the information in the same manner as today. For database evolution, we will work with AWS solution architects, as we do today, to provide skilled resources from both teams during planning and design stages of the new database-as-a-service model.

Table 4-4 defines the expected responsibility assignment matrix (RACI) for application evolution high-level activities across key technology stakeholders for the Analyze and Decouple and Modernize (Design & Implementation) phases.

Stakeholder	Analyze	Design	Implementation	Comments
Accenture	Responsible	Responsible	Responsible	Responsible for the definition of the application evolution approach, design, and implementation.
AWS	Responsible	Responsible	Contributor	Responsible for co-creation with Accenture on approach and design via strategic staffing of AWS solution architects. Contributor during implementation for continuous improvement based on latest knowledge of platform roadmap.
Consortium	Approver, Contributor	Approver, Contributor	Approver, Informed	Approver of analysis, designs, and application evolution SCRs for implementation. Also, a contributor to analysis and design, and will be informed on implementation progress on a frequent basis.
BenefitsCal Contractor, Print Services Contractor, Infrastructure Contractor	Informed	Informed	Informed	Informed about plans, schedule, and progress updates.
Counties	Informed	Informed	Informed	Informed about plans, design, schedule, and regular progress updates. (County reports and County specific applications will be impacted by the database evolution and data migrations)
Infrastructure Contractor	Contributor	Contributor	Contributor	Infrastructure Contractor will be responsible to run scripts related to database and network.

Table 4-4. Our Communications Approach considers all parties in the CalSAWS project.

Ad-Hoc Reporting Capabilities and Ancillary Applications

For county auxiliary applications and reporting impacts, we will maintain the legacy database (which includes the CalSAWS data lake) in sync with the new system while evolving the database until the counties' evolution journey is complete, at which time the legacy database will be retired. **Counties that are not on the CalSAWS data lake at the time of this modernization effort will be able to execute their reports built against the legacy data tables until they are ready to begin using the new tables or the CalSAWS data lake.**

As part of preparing the counties for their migration to the data lake the following items are considered:

- Material will be prepared that will outline the mapping from the legacy to the data lake
- Webinars with the impacted Counties will be held to educate and inform
- As required, the Counties may utilize the County Enhancement Request process to request assistance to complete the migration of their ad hoc reports and database queries tied their ancillary applications.

Benefits of This Approach

We identify the benefits of Accenture's **integrated, incremental, and business-driven approach** to breaking down the application in Table 4-5 for common functionality and decoupling program-unique business logic to create a microservices architecture.

Category	How
Addresses Security Considerations	<ul style="list-style-type: none"> • Improves security as fewer environments will exist • Embeds security measures throughout the lifecycle of each application evolution change • An API-first and event-driven approach to securely break down the application by operational business functions • 100% alignment with the existing AWS architecture, avoiding the introduction of new architectures that must go through rigorous security validation
Reduces Costs	
Improves Optimization	<ul style="list-style-type: none"> • Complete your modernization journey [REDACTED] including preliminary scoping and design of 92 microservices • An integrated approach avoids two full code refactoring efforts, greatly reducing risk • Rapid spin up or ramp down of lower environments as needed through a single step process via microservices deployment patterns integrated into the CI/CD pipeline • Preservation of the API for interfaces • Decommission and removal of middleware components and core databases, using a toggling feature to turn specific functionality on and off <p>[REDACTED]</p> <ul style="list-style-type: none"> • Improved resiliency when one faulty microservice is unavailable, others are unaffected • Utilizes the investment in the creation of the CalSAWS data lake to preserve CalSAWS Reporting/Analytics solution

Category	How
Addresses Scalability and flexibility	<ul style="list-style-type: none"> Microservices facilitate self-contained system changes which enable quicker release cycles and scalability ease delivering business value sooner.

Table 4-5. Benefits of Accenture's approach and how these advantages are realized.

Continuous Improvement and Innovation

As part of our project-wide Continuous Improvement Program (CIP), we will evaluate and implement ongoing improvements to our methodology for breaking down the application. Improvement areas may include speed, quality, cost, security, user experience, and communication effectiveness. The program will run on a quarterly cycle and will be led by our CalSAWS CIP Manager, Sean Swift. At the end of each quarterly cycle, our CIP Manager will work with our M&E Enterprise Architect Luz Esparza, our application modernization team, the Consortium, and the AWS solution architects to:



- Assess opportunities for improvement within application performance throughout the evolution process based on aligned set of KPIs such as transaction completion time
- Evaluate feedback from other contractors on how to continue improving how we manage change
- Review utilization of resources against application evolution and right size environment configuration as needed for cost optimization and performance improvements
- Evaluate AWS and other platform roadmaps for upcoming platform changes. Assess impact of those changes to the application evolution plan
- Conduct quarterly retrospectives to present findings on the current framework and create an action plan of quarterly improvements



To further streamline and improve county operations, we will take a close look at **task management** as we break down that business function by leveraging a user-centric design approach. Task management is unique as it varies by county and requires the project to be fluent in each counties' business process. The best way to do this is by working directly with the counties to identify the best improvements to the existing functionality. Based on these outcomes, we will look for opportunities to streamline task assignment based on demand and availability, how tasks relate to workload

assignment, and opportunities for automation, such as auto-closure.

Refer to the UCD in Action: Improving the Task Management User Experience - Section 4.3 System Change Requests for additional information on how we will embrace a User Centric Design approach to enhancing our modernized Task Management Solution.

4.2.1.2 Tools and Technology

Our solution contains accelerators and tools, described in Table 4-6, that reduce risk, accelerate delivery, and solidify the foundation for lightweight architectures across the Consortium's landscape. These tools will benefit the Consortium by reducing the decoupling effort by approximately 75 percent.

Tool Name	Features and benefits

Tool Name	Features and benefits
Cloud Mover	<ul style="list-style-type: none"> • Cloud-native modernization platform to extract efficient microservices automatically and quickly from complex monolithic applications • Automated and faster application code containerization
vFunction	<ul style="list-style-type: none"> • Assessment, documentation, transformation, and refactoring • Automatically decouples and decomposes the application and validates that business flows remain intact, while the current state application continues to function • Achieves up to 99.9 percent of automation with the lowest risk and minimal business disruption
CAST	<ul style="list-style-type: none"> • Detail level assessment of the application and technology footprint to determine technical dependencies and affinity • Validates top-down and bottom-up approach for domain modelling, dependencies, and reduces business risk

Table 4-6. We utilize leading accelerators, tools, and technology to meet your business needs.

4.2.1.3 Results Delivered

Monolithic Application to microservices transformation for Large U.S.-based Telecom Client



Our Approach in Action:

Our client wanted help to design and build a sustainable microservice system to move away from their monolithic legacy system. We assessed the existing legacy application using Domain Driven Design (DDD) and designed a suite of microservices from one application, enabling each microservice to run its own process and communicate with lightweight mechanisms. These microservices were identified and set up based on business work functions to minimize business impacts and increase business value. We tested the modernized microservices for security with security scans built into the pipeline (SCA, SAST, DAST Scans etc.) and targeted Load Testing in performance environment. We used common seed template for the migration of legacy functionality to microservices which resulted in standardization of code, logging, reuse of common template and promoted CI/CT/CD automated infrastructure.

Results Delivered:

- Improve time to market from several hours to push change in production to deployment at microservices level within minutes.
- Increased production resiliency from single point of failure to robust applications with failover capabilities.
- Improved code quality with focus on quality at each microservices level.
- Reduced operational cost with faster issue resolution and optimized infra spend.

4.2.2 Maintaining Legacy Architecture

Item # ME-UA5

Describe how you will maintain the legacy architecture during evolution, how platforms and how data will be kept in-sync, how changes will integrate with existing technologies and networks, how changes will be tested, and any other factors to be addressed, including security.

As the CalSAWS system moves away from its current monolith architecture to a microservices-based architecture, the legacy and future state architectures will operate simultaneously, and we will continue maintaining the legacy architecture until the to-be state is fully realized. This will confirm that the legacy system is up to date and secure, and that the Counties can continue to support business services without disruption.

4.2.2.1 Our Approach to Maintaining Legacy Architecture During Evolution

Our approach to breaking down the CalSAWS application enables the legacy architecture and new application services to coexist using a **Decoupling Approach**. As described in Section 4.2.1 Breaking Down the Application, the Decoupling Approach enables both the legacy and new functionality to co-exist until the legacy environment is retired.

The flow of a generic request is depicted in Figure 4-14, and runs through the end-to-end process between the various layers. This flow diagram outlines how our solution is balanced and realistic, and thoroughly addresses the required changes. Our approach to decouple and modernize the CalSAWS application includes a Feature Toggle, which is turned on and off during specific parts of the data migration process. Figure 4-14 also includes the Feature Toggle On and Off scenarios and is further described in Table 4-7, with numbers correlating to workflow actions.


Key Success Factors

- Ongoing legacy system maintenance through iterative evolution journey
- Continuous data synchronization between legacy and new
- Security embedded in approach
- Impact-driven test approach

Table 4-7. The Feature Toggle allows us to turn features on and off as the system is decoupled and modernized.

Because the legacy environment will be operational until the migration is complete, it will be maintained to the same standard as it is today. As described in our staffing approach, Accenture will have two teams: one dedicated to maintaining the legacy system, and one dedicated to the modernization effort. With the dedicated legacy team, defects will be promptly remedied, and required changes will be applied to the services that still exist in the legacy environment, using our transformed SDLC approach described in Section 4.3 System Change Requests. In addition, we will support the infrastructure vendor on maintenance activities—such as regression testing of the application to validate platform updates, compliance to security measures, and upgrades to the operating system—maintaining the legacy architecture until the future-state is fully realized. With the dedicated modernization team, we will be focusing on the prioritized list of new services.

Simultaneously re-platforming



430 reports
130 dashboards

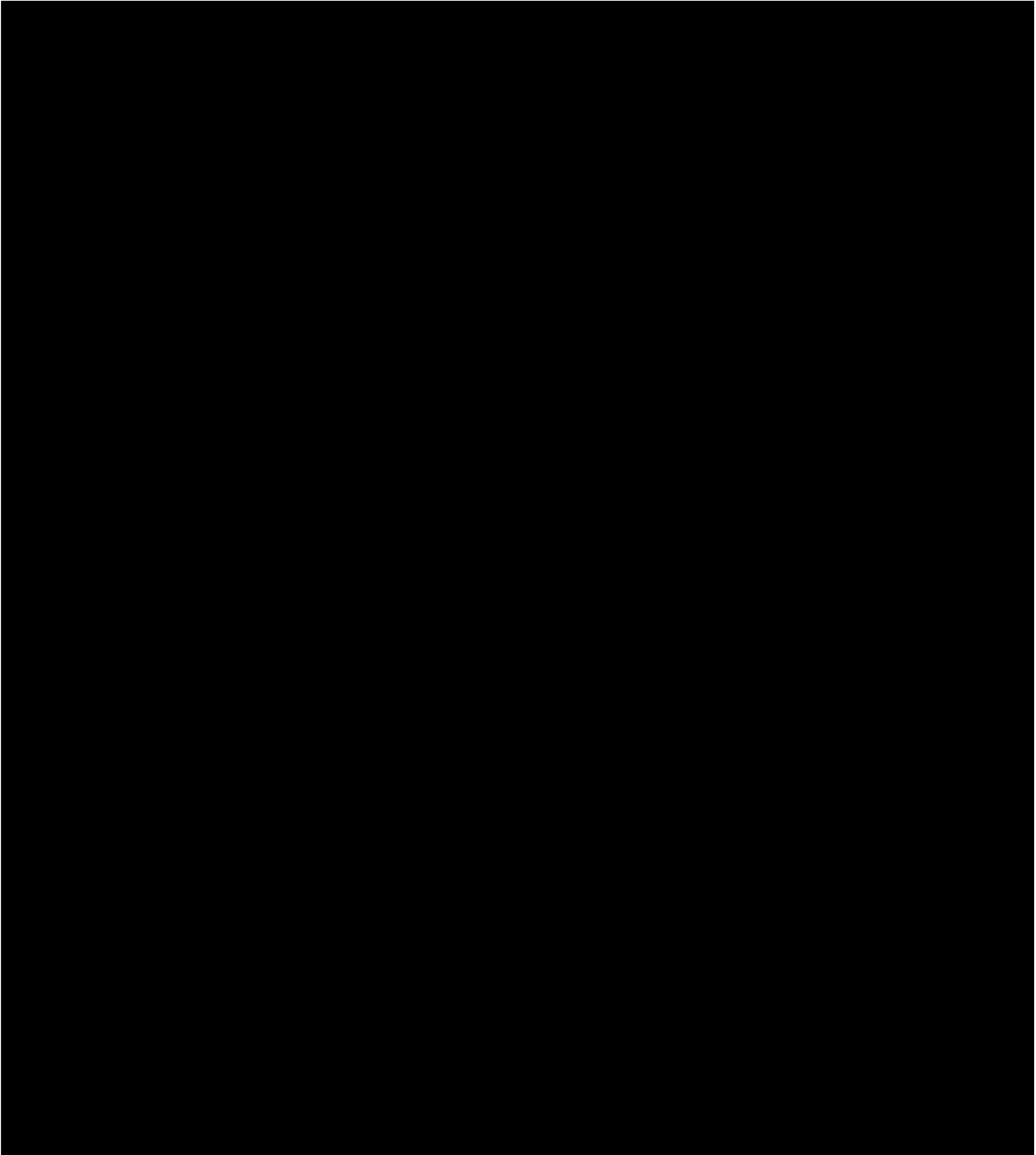
Our decoupling approach and dedicated teams allowed us to successfully manage legacy CalSAWS application dashboards and reports in production while simultaneously re-platforming the same 130 dashboards and 430 reports to a new cloud architecture.

2 CLS I/ME 22.0257

An in depth look at how changes that impact legacy and new services simultaneously will be handled is detailed later in Section 4.2.2.5 Other Factors to Be Addressed. We will add an additional classification for SCRs of Legacy or Modern, so that they can be addressed appropriately and prioritized effectively. To assist with SCR prioritization and scheduling, we will also tag new SCRs to the "broken-down" business function(s) they impact.

We demonstrated the effectiveness of this approach when we re-platformed the legacy CalSAWS reports and dashboard into its current modern, cloud-native state (data lake). We managed both the legacy and new reports architecture until the evolution was completed. During the reports/analytics modernization approach, we maintained both platforms simultaneously. This was achieved by deploying dedicated teams: one focused on

maintaining legacy application dashboards and reports, and the other focused on re-platforming efforts on the new cloud architecture. Legacy reports were reverse-engineered to the new cloud environment, with decommissioning taking place as new reports were deployed in an Agile-like fashion. User profiles and access levels were validated as part of the solution to verify security measures were upheld, and rigorous performance testing was completed as part of the SDLC process prior to releasing reports to the production environment.



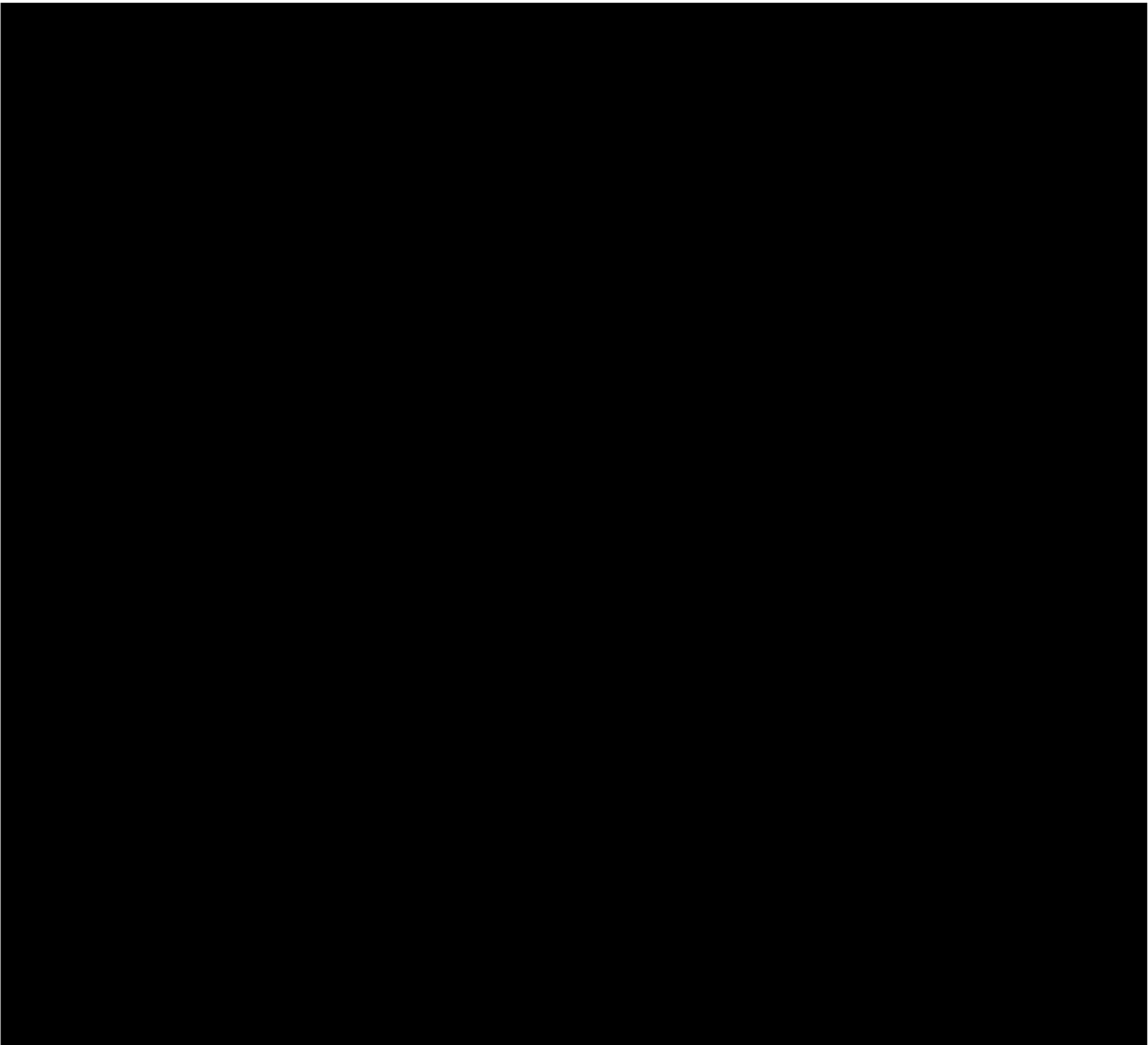


Table 4-8. With the Feature Toggle enabled, solution components are in place to confirm the platform and data are kept in sync.

The data is not required to be stored in the same way or with the same constraints in the microservice and legacy databases. The only requirements are tracking changes to objects and subscribing to events of the objects in use. Table 4-9 describes the challenges to keeping legacy and microservices data in sync.

Challenge	Mitigation Strategy
Data Integrity	It is critical that the data in the legacy data model is kept in sync with the microservice data in near real-time. Our approach to using event streaming will keep the legacy data in sync with the microservice data asynchronously. This will be achieved by automating updates to the legacy data model as they are made to the microservice data model. To confirm this, we have reviewed the payload size for the data that will

Challenge	Mitigation Strategy
	be synchronized, and we anticipated the data will be updated within three to five milliseconds.
Data Conflicts from Different Business Domains	In CalSAWS, there are certain tables that are shared between different business objects. Given this, it is possible for there to be data conflicts if different users are attempting to update the same data from different sources. While the microservices will each have dedicated databases which prevent this issue, there is the potential for data conflicts while maintaining the legacy data structure. To mitigate this, we will develop a conflict resolution process in our event streaming architecture that updates the legacy data.
Data Conflicts from Different Services	As the CalSAWS application is decomposed into microservices, each business object will be independent, eliminating shared tables and data dependencies between different services. If architected improperly with shared data, conflicts can occur if multiple services are attempting to update the same data. By utilizing tools like vFunction, data dependencies will be identified during the analysis phase of the project. This will facilitate the creation of modular databases aligned to the microservices, where there will be no shared tables/database. This will also prevent the creation of conflicting data from multiple sources.

Table 4-9. Our tools and processes keep your data in sync and secure.

Functional Synchronization

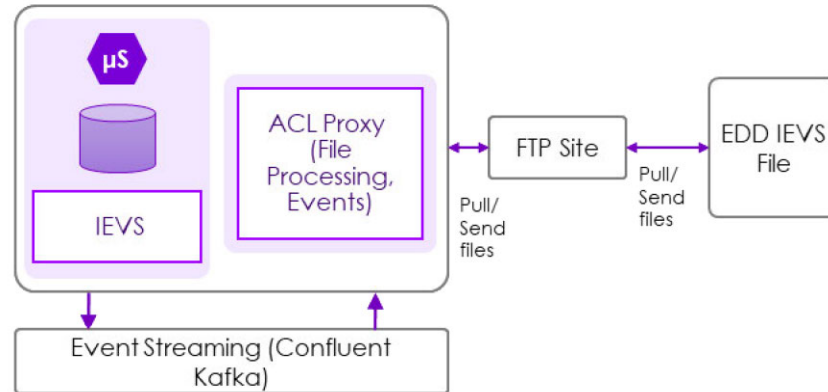
One of the main drivers of using a decoupling approach to modernization, is that it **significantly reduces the need to keep functionality in sync** between legacy and modern environments. This is because the modern services are deployed as soon as they are ready, and once operational the matching legacy functionality can be decommissioned. Compare that to a big bang modernization approach using two parallel tracks year over year, and the same changes (policy and operational) must constantly be made to legacy and modern environments as time passes. To further reduce the need for duplicative updates to modern and legacy work, the team will make every effort to not make changes to legacy functionality that is in the process of being modernized. Should such a change have to occur, and if it is a defect, two SCRS will be logged: one for legacy and one for modern. Should such a change be a SCR, we will follow the process described in Section 4.2.2.5 Other Factors to Be Addressed.

4.2.2.3 How Changes Will Integrate with Existing Technologies and Networks

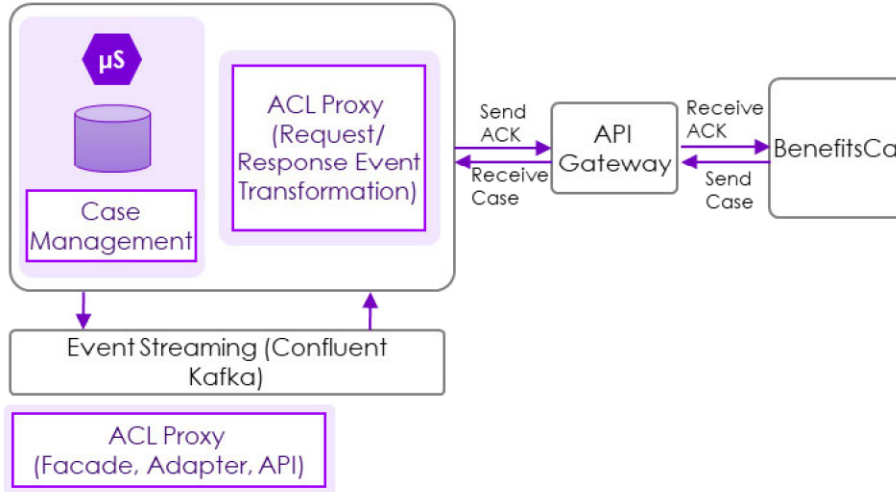
The **API-driven architecture** will facilitate integration of changes with existing systems on the established networks through increased agility and speed in a cloud-native environment. The effort put forth by Accenture CalSAWS to implement API architecture helps insulate the system and supporting networks when changes are made and minimize impacts. All changes will follow the SCR process, with changes fully tested on both the legacy and new architecture prior to release to prevent disruption to current business operations. All technologies we have selected for the evolution are compatible with existing technologies, AWS services, patterns and networks used with the CalSAWS environment.

Anti Corruption Layer

File-based Integration



Real-time Integration



2 CLS MO 23.01.20

Figure 4-16. The [REDACTED] helps prevent corruption and uses bounded contexts and integration protocols to verify non-desirable architecture characteristics will not be inherited into the new system.

To demonstrate how the ACL will work, let's look at an example of a file-based interface being read into the system and an outbound API.

IEVS File Based Interface – Income and Eligibility Verification Service (IEVS) is a business domain in CalSAWS that will be decomposed from the legacy monolith into a microservice. It is populated from a flat file interface that is received. As this flat file is read into CalSAWS, the data will be processed and validated through the ACL layer to maintain the data integrity of the microservice.

BenefitsCal API – BenefitsCal is a business domain within CalSAWS that integrates the BenefitsCal Self-Service Portal with CalSAWS. As the CalSAWS system is decomposed, this will become a separate microservice. There is an exposed API that integrates with this business domain with BenefitsCal allowing it to display information directly from CalSAWS. This could include customer information or SAR 7 data as an example. As this data is being transmitted through the API, it will be processed by ACL to ensure that the data conforms to the appropriate format as defined by the API ensuring there is no impact in the integration with BenefitsCal as we decompose the application into microservices.

4.2.2.4 How Changes Will Be Tested

Testing Newly Migrated Services and Validating Legacy

During the Solution Planning phase, we will define the appropriate test approach for each application evolution SCR. We will conduct testing against the system where the changes were made—legacy, new, or both—depending on the approach of the change and apply all test methods mentioned, based on the intended impact of the change.

- **Sprint testing using our Hybrid-Agile approach** will verify that the change was completed as expected by the definition of the SCR. We will apply this test type to all application evolution changes, and it will continuously occur with every sprint, not just at the completion of the evolution SCR. This reduces the need for defect re-work and re-testing, greatly reducing the time it takes to implement the change and increasing stability to the application by catching issues earlier in the development lifecycle.
- **Automated regression testing of the legacy and modern service** to make sure there are no impacts to the end-users and the CalSAWS business process by utilizing our existing automated regression testing framework to test each Application Evolution SCR. Our team's extensive experience and application and integrated eligibility knowledge allows us to quickly iterate on and enhance our existing automated regression tests to efficiently validate modern services. Where the change has no impact to end-users, modernized changes can be rapidly tested using our existing Automated Regression Testing framework.
- **Manual testing of new modern service** to validate requirements of the design that cannot be tested in an automated manner.
- **Manual regression testing** of the legacy system for any possible adverse effects of the deployment of the new service on any other areas of the application that cannot be automated. Manual regression test suite may be more targeted or broad depending on the change impacts and dependencies for each application evolution change.
- **Security testing** to verify that new vulnerabilities are not introduced based on the changes on an on-going basis. We will apply this test type to all application evolution changes throughout the development lifecycle.
- **Performance testing** to verify that the application performance is as expected following the modern service changes to all application evolution changes.
- **User acceptance testing** will validate that the intended use of the system by end users was not altered by the change. We will apply this test type as needed to application evolution changes.

The overall benefit of this testing approach is to minimize business disruption, catching items as early in the process as possible. This allows us to continue to maintain CalSAWS while evolving the architecture to a technically viable steady state.

4.2.2.5 Other Factors to Be Addressed

System Change Requests That Occur During Evolution

As described in Section 4.3 System Change Requests, we will conduct a Solution Planning phase for each SCR. During this phase, we will determine if an SCR should be implemented against legacy, target or both architectures by evaluating the release timing against the evolution roadmap. As shown in Figure 4-17, we will encounter situations where we are in progress with evolving a business area while the core M&E team is assigned a priority SCR to implement simultaneously. In such situations, we will implement the change in both the legacy and new architecture following the defined process for implementation and testing.

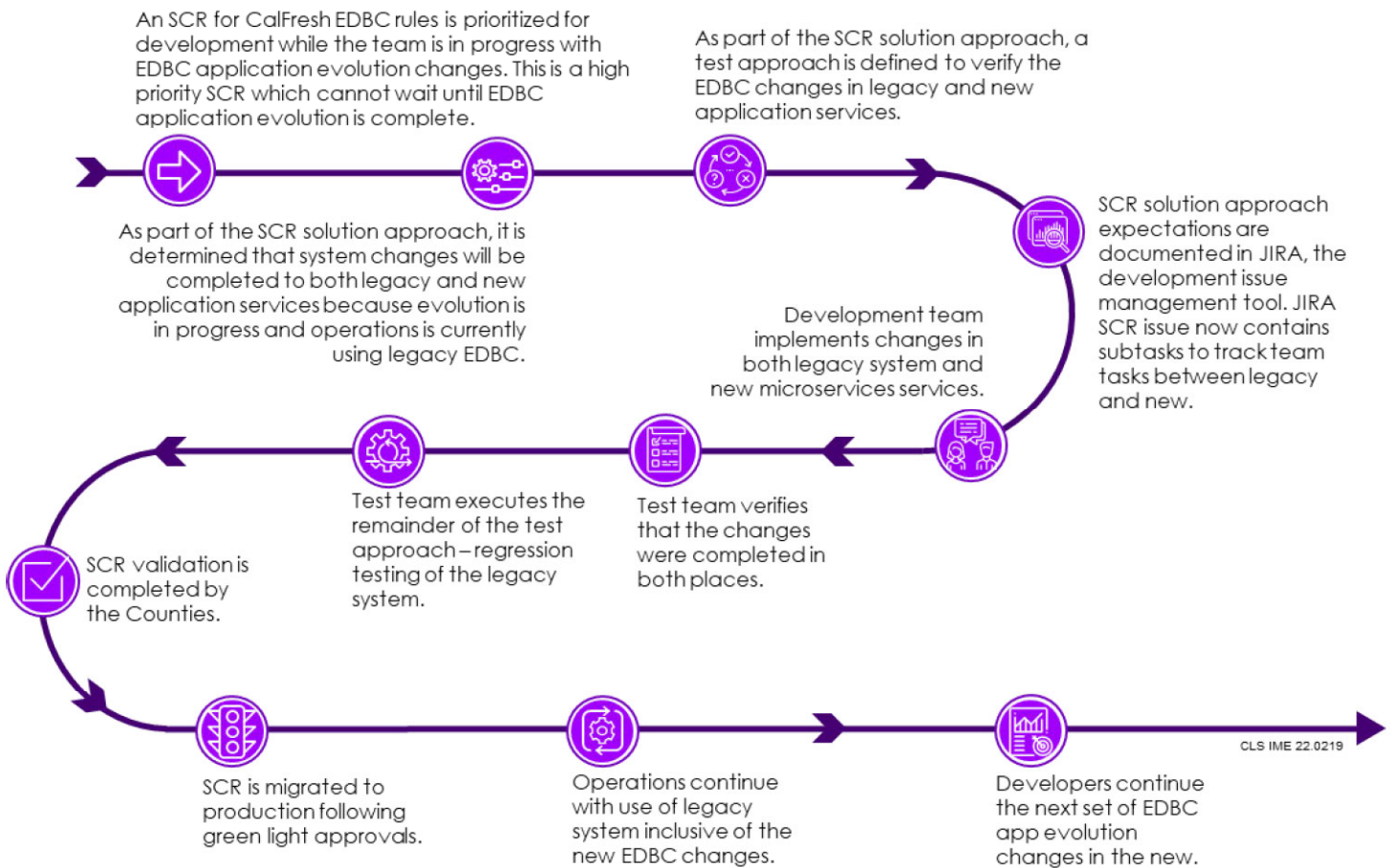


Figure 4-17. How we will address necessary SCRs that impact both legacy and modern services.

Security

Accenture will continue performing individual application security tests for the legacy architecture application code, as well as integrate security within the DevSecOps pipeline by automating a continuous integration/continuous delivery (CI/CD) pipeline that accesses code, logic, and application inputs to help detect CalSAWS software vulnerabilities and threats.

We will continue to provide additional security services such as identity and access management, privileged access management, governance, risk, compliance, and data privacy, as the application evolves to the future state.

4.2.2.6 Benefits of this Approach:

- A blend of data preservation, incremental value delivery, and risk mitigation
- Ability to operate the legacy and new architectures simultaneously with an iterative switch from legacy to new by business domain
- Reduces the risk of business operations disruption and brings value early and throughout the evolution journey
- Provides continuous system enhancement
- Enables you to reduce or remove core databases and middleware components and to optimize scalability and application longevity
- Users will be performing "business as usual" while net new architecture is developed, maintaining quality services for Californians while evolution is in flight
- Comprehensively architected modular system completed in a shorter timeframe

4.2.2.7 Results Delivered

Reports and analytics functionality re-platform for CalSAWS



Our Approach in Action:

Accenture assisted the Consortium in their request to re-platform approximately 550 legacy dashboards and reports from an Oracle solution to a new cloud-based architecture. This was completed in a two-year timeframe using an Agile methodology and the SDLC approach, in collaboration with various stakeholders and user groups. An Agile-like release schedule was implemented to allow reports to be re-platformed over time, and to support development per release.

Results Delivered:

- Developed a Proof of Concept to identify the best solution for CalSAWS and created an inventory of Oracle business intelligence legacy reports to determine artifacts that required re-platforming.
- Legacy reports were reverse engineered to verify cloud compatibility, informing subsequent report design and development.
- Thorough assembly, system, and client testing was conducted prior to a production soft launch to a subset of users, and thereafter, a hard launch of reports was deployed to production to all users.

Maintaining monolithic application for Large U.S.-based Telecom Client



Our Approach in Action:

Our client wanted to transition away from a monolithic application to a nimbler microservice architecture and was concerned about maintaining the monolithic application while starting the new approach. The syncing of data between the old and new platforms was a priority to prevent any data mismatches or delayed updates. Accenture proposed an automated deployment approach to manage the legacy and microservice architecture and event-driven architecture to integrate changes between business functions so that dependent services and external contractors were not impacted. By using an API Gateway and Anti-corruption Layer, we detached the monolith from legacy technologies and revealed new entities and functions that could be decomposed.

Results Delivered:

- The transition to a microservice architecture enabled the regression testing of the application and reduced the likelihood of a large impact to the overall application.