White Paper

Modernize the Monolith

A Guide to Modernizing Legacy Systems for Digital Transformation

pivotree™

Executive Summary

Digital commerce has gained tremendous momentum in the past two decades largely owing to technological advancements and the impetus of evolving customer expectations. Companies, under extreme pressure to keep pace, face a critical issue: they are forced to operate with legacy systems that were expensive to implement, expensive to maintain, and difficult to abandon, yet are inherently unable to meet customers' growing requirements for speed and agility.

Given the large capital expense of these systems and current economic conditions, a systemic overhaul of the monolith is not a feasible option for most.

The good news is there's another option. With the emergence of composable commerce microservices and integrated solutions, it is now possible to modernize your solution without having to get rid of your legacy system or spend millions of dollars in new technology investment.

The growing recognition of the importance of modernizing existing monolithic systems with integrated solutions is evident through the increasing adoption of advanced commerce solutions. This approach affords organizations the opportunity to drive agility and responsiveness in their entire ecosystem, without undergoing costly and disruptive overhauls. By gradually replacing outdated functionalities with modernized counterparts that align with current objectives, businesses can harness the benefits of a modernized monolithic system powered by composable commerce. The integration of these systems within the modernization process empowers organizations to enhance operational efficiency, improve customer experiences, and remain competitive in today's dynamic business landscape.

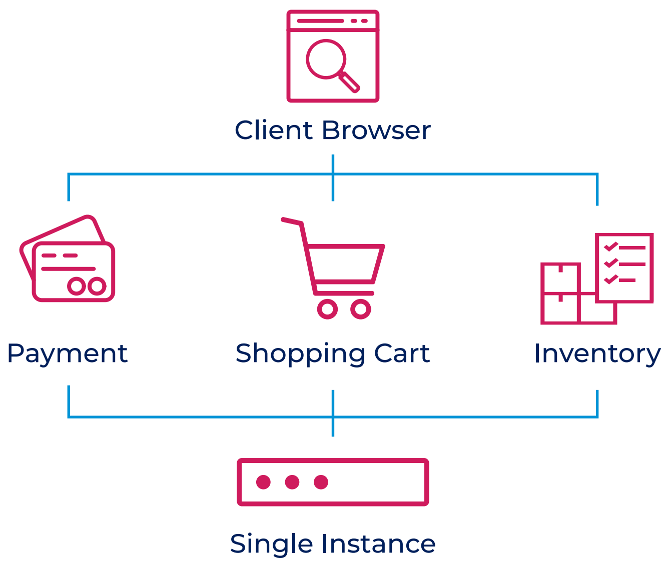
Modernizing existing systems allows companies to drive agility and responsiveness in their commerce ecosystem, without high costs and disruption.

Defining the Monolith

What is Monolithic Architecture?

Monolithic architecture refers to the traditional software model that is built as a single "monolithic" unit that is independent of other applications. A monolith is large, glacial, indivisible, and singular. Monolithic systems are pretty much the same – large computer networks built on a single code base that addresses multiple business objectives in a single system. Making changes to any aspect of these systems requires the entire stack to be updated by tweaking the code base, building in the changes, and deploying the update on the service side. In simple terms, any change to be made in the system requires the whole machinery to stop, rebuild, and reboot.

MONOLITHIC ARCHITECTURE



As a recent Forrester report states, "Most legacy core software systems are too inflexible, outdated, and brittle to give businesses the flexibility they need to win, serve, and retain customers."[1] The leading cause of this inflexibility and brittleness that plagues most legacy systems is the use of monolithic architecture. These systems are characterized by millions of lines of code with multiple interdependencies therein. These applications pose a few fundamental challenges that make it challenging for businesses to adapt to evolving customer needs.

The Challenges of Managing a Monolith

- These systems get too large with time and become increasingly difficult to manage
- Even a small change requires the redeployment of the whole application
- As the application size increases, the time to start up and deploy also tends to increase
- New developers find it hard to understand the architecture and logic of the application, even if their responsibility is limited to a single functionality
- Increased load or traffic on any single facet of the application, instances of the entire application are deployed across multiple servers. This is inefficient and resource sensitive
- Adoption of new technology is challenging as it impacts the entire application in terms of cost and time
- A single bug in any module can negatively impact the entire application

The Monolith Conundrum:

- Customization creates complications
- Slow response to market trends
- A single point of failure due to high dependency
- Endless testing before taking updates live

Add all the above complications and you have a big hindrance in delivering on the customer promise. Even one of these elements can lead to delays and inaccuracies that, in turn, can lead to higher product returns and decreased customer retention.

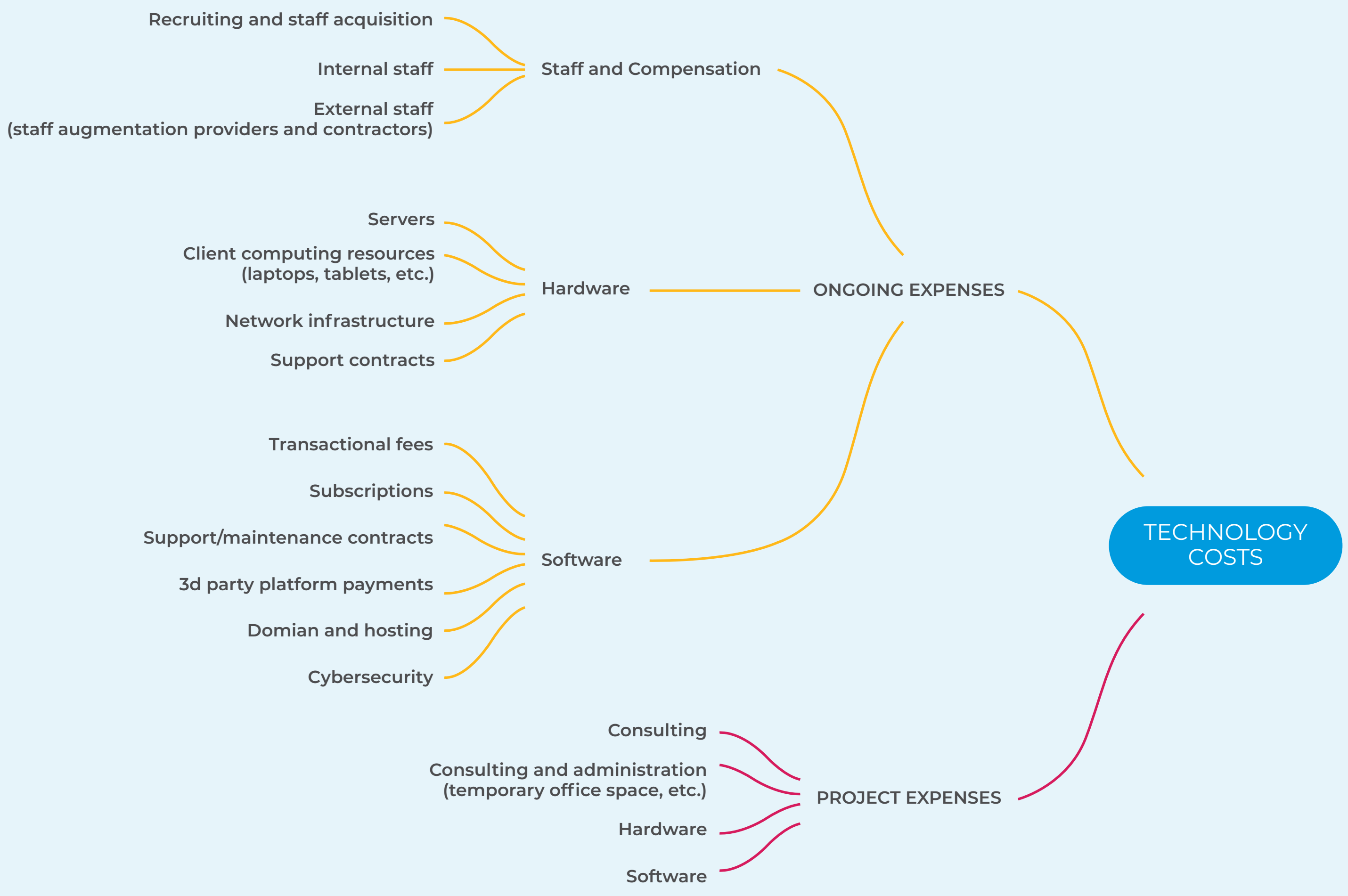
Why Modernize?

Although initially appealing for its ability to expedite application development, a monolithic architecture eventually presents challenges. As businesses expand and user demands evolve, customers anticipate enhanced user experiences, while integration needs intensify. Unfortunately, the monolithic approach hinders growth by creating bottlenecks. Common problems associated with monolithic applications involve extended time to market, a steep learning curve for new developers, excessive dependencies, and protracted deployment periods.

Monolithic systems for commerce vary in capabilities, functionalities, and size, but the one thing that's common with all monolithic systems is cost. The huge investment in monolithic systems usually takes a while to yield a return on investment (ROI), and this is the reason why it doesn't make sense for businesses to consider ditching monolithic systems that they have already spent a considerable amount on.

Then again, the cost implications of monolithic systems are not restricted to the initial investment alone. Overhauling a monolithic system also leads to the need for a restructuring of the smaller moving parts that drive the larger machine. The below mindmap is a great example of this. Change the existing system, and the trickle-down effect means that you will have to once again spend time and resources finding the right talent and training the team. Similar impacts may also manifest in hardware and software requirements, and project expenses.

The Costs of System Overhaul


Recruiting and staff acquisition
Internal staff
External staff (staff augmentation providers and contractors)
Staff and Compensation
Servers
Client computing resources (laptops, tablets, etc.)
Network infrastructure
Support contracts
Hardware
ONGOING EXPENSES
Transactional fees
Subscriptions
Support/maintenance contracts
3d party platform payments
Domian and hosting
Cybersecurity
Software
TECHNOLOGY COSTS
Consulting
Consulting and administration (temporary office space, etc.)
Hardware
Software
PROJECT EXPENSES

Leveraging Composable Commerce with Integrated Solutions

According to Microsoft, “A microservices architecture consists of a collection of small, autonomous services. Each service is self-contained and should implement a single business capability.”[2] While the definition refers to these applications as “small,” this may not always be the case. The key characteristic of microservices-based applications is that they are independent and coupled with specific functionality. The codebase for each microservice is different to perform only a single task, with each microservice communicating with the other through APIs (application programming interfaces).

The key characteristic of microservices-based applications is that they are independent, making your system architecture more agile, scalable, and improving time-to-value.

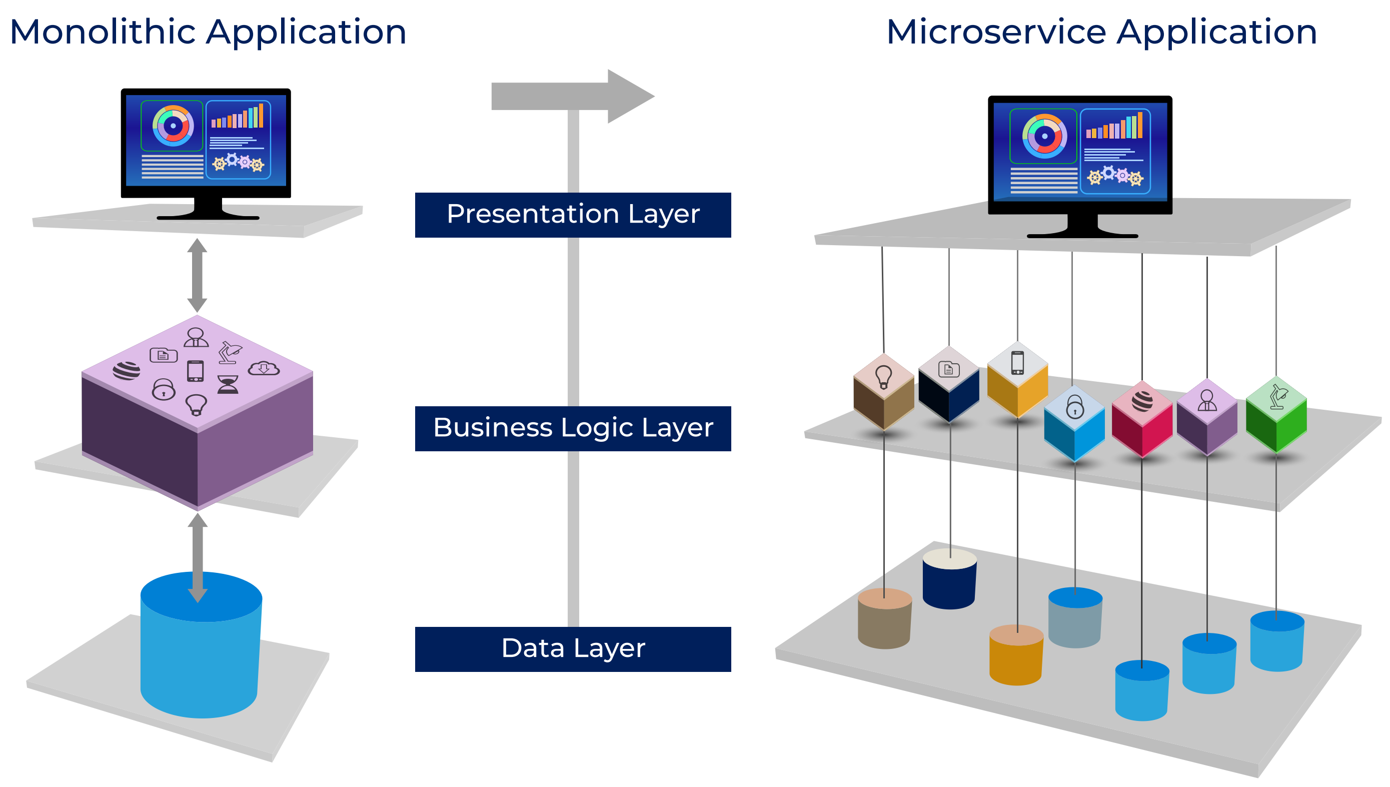
What You Get with Microservices-Based Composable Architecture

AGILITY: The agility of microservices lies in their small and independent nature, allowing for swift updates to address new requirements without impacting the entire application. This agility also enables businesses to solve point problems with point solutions without the baggage of unneeded technical debt.

SCALABILITY: Microservices offer independent scalability. Each microservice can be scaled individually, providing cost savings in cloud environments by avoiding the need to scale the entire application.

EASY MAINTENANCE: Microservices' small and specialized nature makes maintenance significantly easier. This simplicity enables a small team of developers to handle the maintenance tasks effectively.

TIME TO VALUE: These capabilities can be implemented and utilized incrementally as opposed to a ‘Big Bang’ approach, ensuring that businesses can derive maximum value from the get-go.



A Planned Approach to Migration

Composable architecture gives businesses the flexibility to gradually cycle out pre-existing functionalities for modernized ones that are better suited to your business objectives. Migrating to these solutions, however, is an incremental process rather than a one-and-done overhaul.

GAIN A THOROUGH UNDERSTANDING OF THE EXISTING SYSTEM

Begin by thoroughly analyzing the existing monolithic system. Gain a clear understanding of its architecture, components, dependencies, and data flows. This step is crucial to identify potential integration points and challenges.

DEFINE INTEGRATION GOALS

Determine the specific objectives you want to achieve through the integration. This could include improving scalability, agility, or flexibility, as well as enabling new features or capabilities. Clearly define the desired outcomes to guide the integration process.

IDENTIFY SERVICE BOUNDARIES

Identify potential boundaries within the monolithic system where you can extract discrete functionalities or modules that can be encapsulated as services. This step involves breaking down the monolithic architecture into more granular components that can be independently managed and deployed.

DESIGN COMPOSABLE ARCHITECTURE

Define a composable architecture that aligns with your integration goals. This architecture should include the core building blocks of a composable solution, such as modular services, API contracts, event-driven communication patterns, and a central integration layer.

PRIORITIZE INTEGRATION POINTS

Determine which components of the monolithic system should be integrated first based on their criticality, complexity, or potential business value. Start with smaller, less complex modules that can be more easily extracted and decoupled from the monolith.

EXTRACT SERVICES

Extract the identified components from the monolithic system and convert them into modular services. Containerize these services to facilitate independent deployment and scalability.

IMPLEMENT API CONTRACTS

Define clear API contracts for the services to ensure compatibility and consistency. This allows different components to communicate effectively and enables loose coupling between services.

ESTABLISH INTEGRATION LAYER

Set up an integration layer or service bus that facilitates communication between the monolithic system and the newly created services. This layer can handle message routing, event-driven communication, and data transformation between the legacy system and the composable components.

IMPLEMENT INTEGRATION PATTERNS

Apply appropriate integration patterns to enable seamless communication and synchronization between the monolithic system and the composable components.

TEST AND VALIDATE

Thoroughly test the integration to ensure proper functionality, performance, and reliability. Use both unit tests and end-to-end tests to validate the integration and identify any issues or bottlenecks.

PLAN A PHASED APPROACH

For migrating functionality from the monolithic system to the composable solution start with less critical or standalone features and gradually move towards more complex modules. Monitor the migration process closely to mitigate any potential risks or disruptions.

MONITOR AND ITERATE

Continuously monitor the integrated system and collect feedback from users and stakeholders. Iterate and improve the composable solution based on insights gained, addressing any performance or functionality issues that arise.

Modernize With The Help of Professional and Managed Services

Monolithic systems face significant challenges in today's technology landscape, primarily due to scalability limitations, inflexibility in adapting to changing requirements, and vulnerability to single-point failures that can disrupt entire applications. They often struggle with integrating new technologies and frameworks, hampering agility and innovation. Large development teams find coordination difficult, impacting productivity and deployment efficiency. Moreover, the accumulation of technical debt makes maintenance costly and complex over time.

Microservices and integrated solutions, offer businesses the flexibility to gradually replace outdated functionalities with modernized ones that align with current objectives. Without making any ground-breaking changes to the monolith, this approach gives organizations scalability, agility, and the luxury of easier maintenance.

Pivotree Professional Services play a crucial role in modernizing your ecosystem by offering tailored consultancy and implementation services. Whether it's migrating from monolithic architectures to microservices, adopting cloud-native solutions, or enhancing cybersecurity measures, we provide expert guidance and hands-on support throughout the transformation process. Our services include project planning, architecture design, implementation, and post-deployment support to ensure a smooth transition and optimized performance of your technology stack.

Pivotree Managed Services provide ongoing operational support and management for your ecosystem. This includes monitoring the performance and health of your technology systems, ensuring they operate efficiently and securely and serve you optimally through routine tasks such as updates, patches, and backups. We also offer proactive troubleshooting and issue resolution, minimizing downtime and maintaining continuity - all with an eye to mitigating risks and optimizing costs.

Realize Your Vision with Pivotree, the Commerce Experts

30+ years experience

200+ global customers

Expertise in the world's leading platforms and systems

Pivotree has worked with many organizations to streamline operations, while ensuring modern capabilities and cost efficiencies are achieved. With a wealth of experience, a platform-agnostic approach, and a proven track record, we can guide you through an assessment of your ecosystem.

Contact us today for a personalized assessment of your digital ecosystem.

sales@pivotree.com
877-767-5577 | pivotree.com

[1] https://www.forrester.com/report/MODERNIZE-CORE-APPLICATIONS-WITH-CLOUD/RES144356

[2] https://learn.microsoft.com/en-us/azure/architecture/guide/architecture- styles/microservices