AWS Prescriptive Guidance

Strategy for modernizing applications in the AWS Cloud

AWS Prescriptive Guidance: Strategy for modernizing applications in the AWS Cloud

Table of Contents

Strategy for modernizing applications in the AWS Cloud

Vijay Thumma, Amazon Web Services (AWS)

December 2020 (document history (p. 12))

A successful application modernization strategy starts with the business need in mind, and then focuses on technologies. As the journey to the cloud gathers pace, organizations have been looking for ways to accelerate cloud adoption and for a prescriptive approach to application modernization. Amazon Web Services (AWS) approaches application modernization by dividing the modernization roadmap into discrete increments that focus on three phases: assess, modernize, and manage. This article discusses the strategy for assessing and modernizing applications, and is based on the AWS Professional Services team's years of experience helping enterprise AWS customers in their cloud adoption and application modernization projects.

This strategy is for IT and business executives, program and project managers, product owners, and operations and infrastructure managers who are planning to modernize their applications in the AWS Cloud. It explains how to identify mission-critical applications, how to evaluate different modernization approaches (such as refactor, rearchitect, or rewrite), and how applications would benefit from improved scalability, performance, security, and reliability.

The strategy is part of a content series that covers the application modernization approach recommended by AWS. The series also includes:

- Evaluating modernization readiness for applications in the AWS Cloud
- Phased approach to modernizing applications in the AWS Cloud
- Decomposing monoliths into microservices
- Integrating microservices by using AWS serverless services
- Enabling data persistence in microservices

Overview

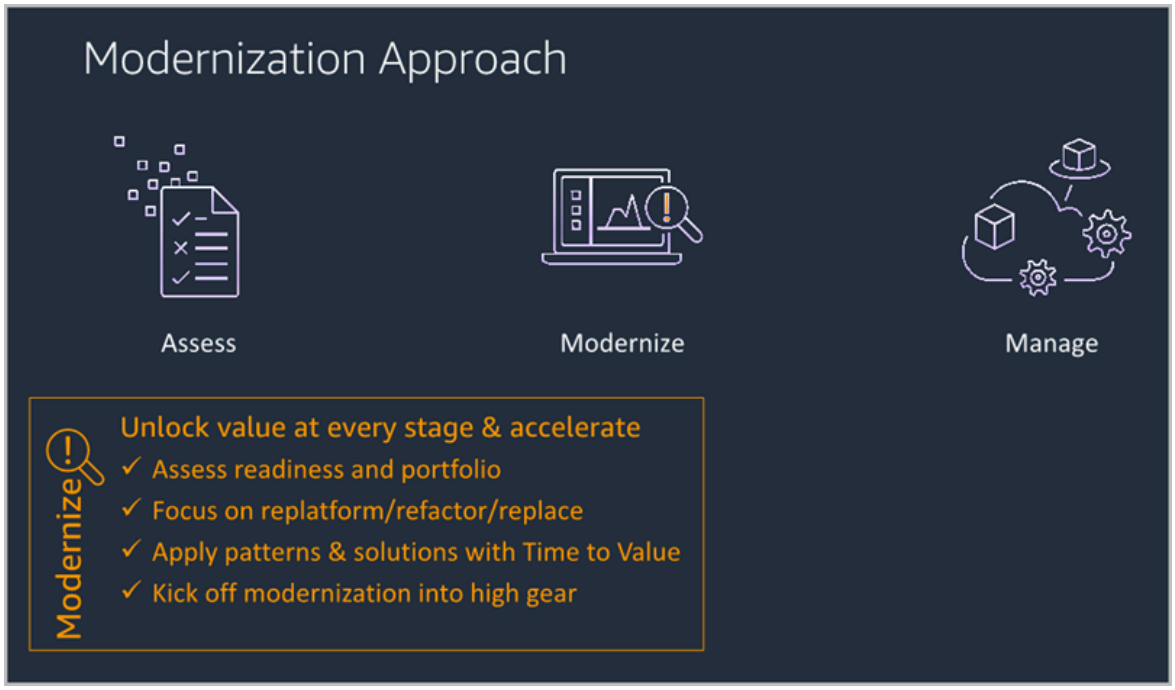
Modernizing your applications helps you reduce costs, gain efficiencies, and make the most of your existing investments. It involves a multi-dimensional approach to adopt and use new technology, to deliver portfolio, application, and infrastructure value faster, and to position your organization to scale at an optimal price. After you optimize your applications, you must operate in that new, modernized model without disruption to simplify your business operations, architecture, and overall engineering practices.

Migrating applications to AWS by using the rehosting (lift and shift) approach doesn't automatically give you the benefits of the elasticity, resiliency, ease of deployment and management, and flexibility that AWS offers. Nor does it automatically modernize your culture and processes to enable high-performing software development. Modernization means taking your application environment in the form that it's in today (most likely, legacy and monolithic) and transforming it into something that is more agile, elastic, and highly available. In doing so, you can transform your business into a modern enterprise.

To optimize your cloud adoption and migration, you must first assess and evaluate your enterprise for readiness. After you assess the readiness of your organization, you can:

- Select one or two applications.
- Modernize those applications so that you can maintain, extend, deploy, and manage them in a way that meets the current and future needs of your business.
- Establish a foundation for modernization at scale through the hands-on experience you gained in the previous two steps. In this phase, you can create a complete modernization solution by determining the supporting infrastructure, application middleware, middleware services (such as databases, queuing software, integration software, and other technologies), and other components.

The iterative approach to application modernization discussed in this article can be divided into three high-level phases: assess, modernize, and manage. These phases are discussed in more detail later in this article.



Targeted business outcomes

Enterprise-scale application modernization requires a holistic approach (assess, modernize, manage) to bind multiple dimensions to provide completeness at an accelerated pace. The framework recommended by AWS envisions modernization across five technical domains: automation, developer workflows, self-service data, architecture evolution, and organization for value. These domains are discussed in more detail in the Strategic dimensions of modernization (p. 4) section. The framework that you can use in your AWS Professional Services and AWS Partner engagements includes a knowledge base with solutions, self-service technical patterns, playbooks, and templates.

A successful modernization project helps produce the following business outcomes:

- **Business agility** – The effectiveness within the business to translate business needs into requirements. How responsive the delivery organization is to business requests, and how much control the business has in releasing functionality into production environments.
- **Organizational agility** – Delivery processes that include agile methodologies and DevOps ceremonies, and support clear role assignments and overall collaboration and communication across the organization.
- **Engineering effectiveness** – Improvements in quality assurance, testing, continuous integration and continuous delivery (CI/CD), configuration management, application design, and source code management.

Achieving these business outcomes requires a holistic approach and a modernization process that's based on a set of strategic dimensions.

Holistic approach to modernization

The journey to application modernization is an incremental effort that involves:

- Making data-driven decisions to analyze legacy and cloud workloads.
- Evaluating processes to move to the cloud.
- Integrating new functionalities such as containers, serverless technologies, and modern databases to support emerging technologies such as artificial intelligence (AI), Internet of Things (IoT), and machine learning (ML).

Continuous modernization across all areas of the organization is the key to success. To get the full value of modernization, your strategy should focus on understanding choices and tradeoffs, and the ability to combine and connect enterprise, differentiated, undifferentiated, and commodity applications. The process begins with an application assessment to align to business outcomes, and allows enterprises to deploy and manage applications optimally.

Today's enterprises might be unable to adapt to new and changing business models if their legacy systems include complexities and inefficiencies that result in the following:

- Lack of agility, where they cannot react quickly to changing business and market demands.
- Lack of flexibility, where they cannot make necessary changes to applications.

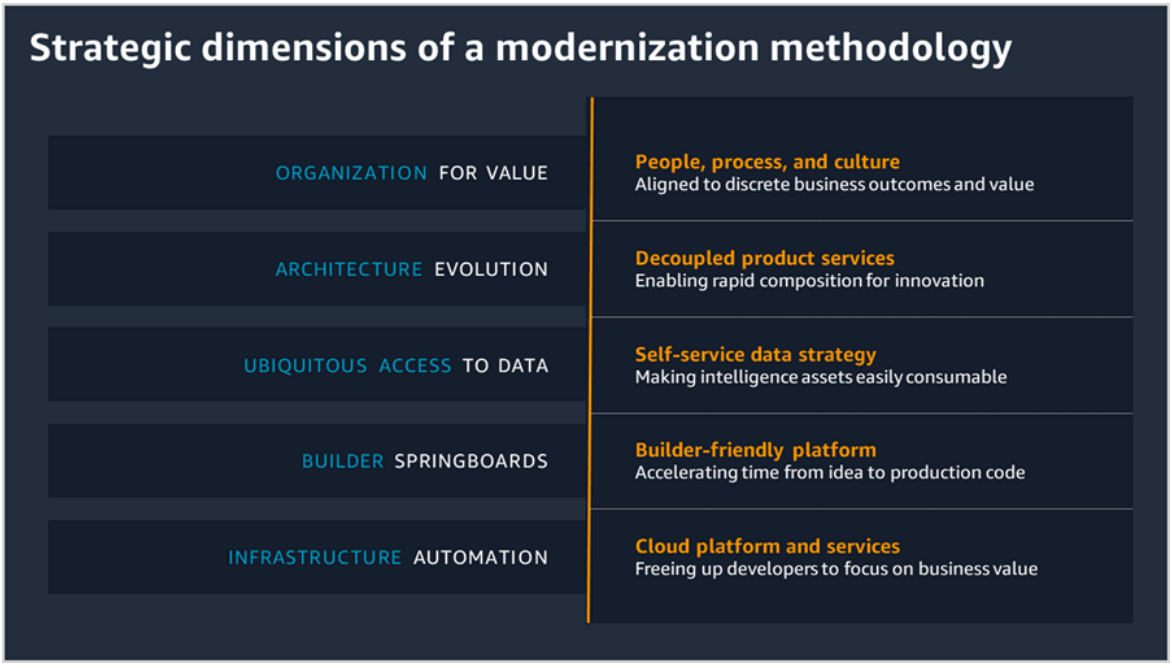
- Lack of scalability, where they cannot introduce new application features or extend existing features that involve new users or capacity.
- Performance issues, when the application doesn't perform to desired standards and metrics.
- Lack of data insights, when too many data silos exist and slow digital innovation.
- Heightened security risks, when applications have gaps and vulnerabilities that don't exist within newer application frameworks where security is built in and integrated throughout.
- Inability to add new applications and services, which impedes the adoption of new technologies and modern architectures.
- Higher costs, because legacy applications and application frameworks often consume more resources, and often create more redundancies and inefficiencies than modernized applications.

Strategic dimensions of modernization

Modern applications provide multi-dimensional benefits to customers when they're developed and managed effectively. You can establish a process for continuous modernization based on a set of strategic dimensions to accelerate innovation by increasing agility, resiliency, and engineering efficiency. By continuously following and building on these proven patterns and techniques, you can deploy existing application components to a modern deployment platform, make existing functionality accessible to new applications, and update application architecture to a fully modern stack.

These modernization dimensions, as shown in the following diagram, are:

- **Organization for value** – Realign organizational structures, governance, and processes to center around small, full-stack product teams that can deliver business value through customer outcomes.
- **Architectural evolution** – Build digital product platforms by moving core business capabilities out of monolithic applications and into a decoupled collection of independently maintainable, evolvable, reusable services that developers can use as building blocks to innovate.
- **Ubiquitous access to data** – Combine modern data architecture, storage, and access patterns with AWS services to allow developers, data scientists, and business users to easily tap into the organizational data stream.
- **Builder springboards** – Bring together a collection of agile software engineering practices (such as DevOps, test automation, CI/CD, and observability), associated tooling, and application layer services into an integrated developer workflow. This workflow defines a path for development and reduces the amount of time to move code from idea to production.
- **Infrastructure automation** – Use a combination of AWS services to create a lightweight infrastructure foundation. Make use of containers and AI/ML to abstract and automate often used infrastructure primitives. This frees up development resources so you can focus on delivering business value through the creation of new products and services for your customers.



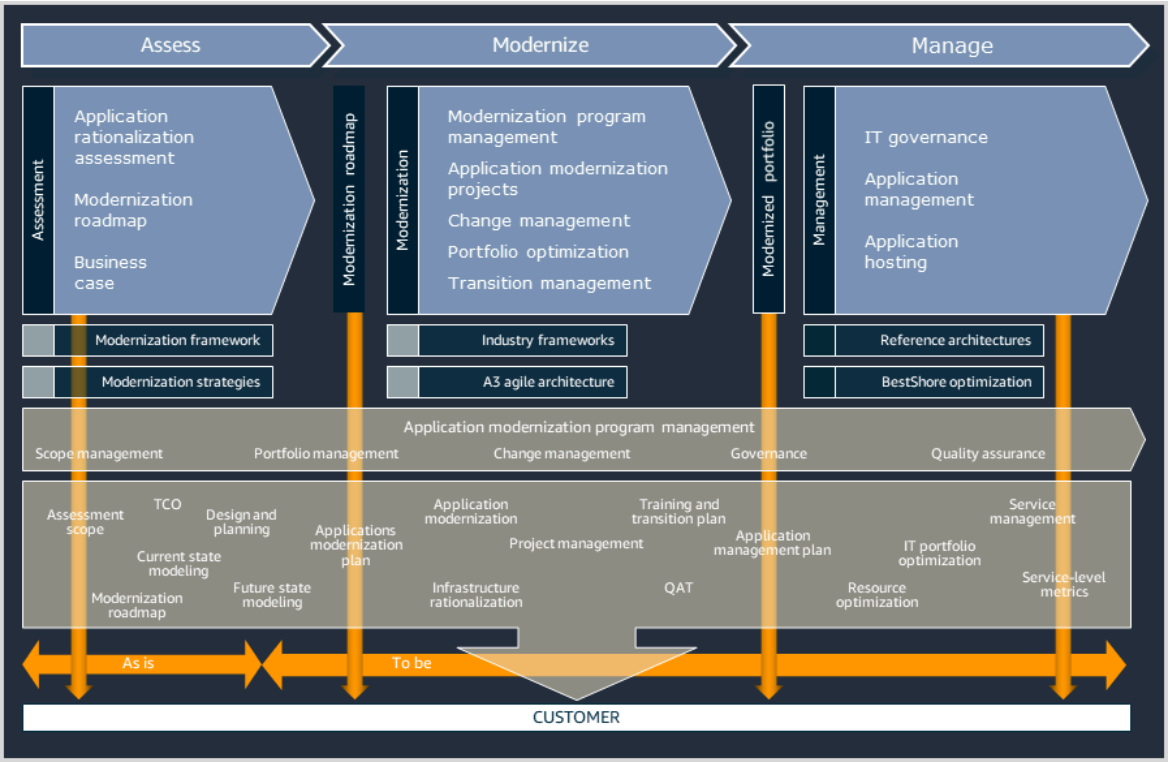
By applying these strategic dimensions of modernization, your organization can become more productive and can deliver measurable and sustainable outcomes. Your organization will be able to:

- Improve and create a differentiated customer experience.
- Accelerate innovation, reduce time to market, and release new products frequently.
- Optimize and avoid costs by spending less on IT infrastructure.
- Increase agility, add new features and functionalities at scale.
- Increase staff productivity by deploying new features faster.
- Improve service-level agreements (SLAs) and reduce unplanned outages.

Phases of the modernization process

Modernizing your applications to meet customer demands and to take advantage of the changing technology landscape is critical to maintaining your organization's competitive advantage and market share. A key strategy to meeting these business demands is to deliver both continued use and real value by converting aging applications to more modern architectures. Having a comprehensive understanding of the application's details and its interrelationships with other systems is a critical step in performing application modernization.

The AWS approach to application modernization is iterative, and can be divided into three high-level phases—assess, modernize, and manage—as illustrated in the following diagram.



The following sections discuss each phase in detail.

Topics

Assess

The first step in an organization's modernization journey is to analyze the existing application portfolio, assess the systems that need to be modernized, and identify the technical solutions required for application modernization. In this phase, you can use the application modernization questionnaire to assess and rationalize the applications portfolio and determine the business, functional, technical and

financial significance (the strategic value) of applications in the portfolio. This will determine how well the organization can support the future state architecture, when it's built.

Activities

- Assess applications through five lenses:
  - Strategic or business fit
  - Functional adequacy
  - Technical adequacy
  - Financial fit
  - Digital readiness
- Group, rank, and sequence applications.
- Document target and interim operating models.
- Understand key technology and regulatory requirements.
- Determine applications that need extensive data migration.
- Clarify the scope and volume of data to be converted.

Outcomes

- Application modernization blueprint
- Technical and functional architecture for the target state for one or two applications
  - Strategic or business fit
  - Functional adequacy
  - Technical adequacy
  - Financial fit
  - Digital readiness

How-to guide

- Evaluating modernization readiness for applications in the AWS Cloud

Modernize

During this phase, you determine project goals and resource requirements, and you build out the implementation roadmap. The goal is to revitalize your applications by using a modernization program that creates a modern, agile application architecture.

Activities

- Determine the milestones for converting your applications' source code and data.
- Complete the mapping of all operational areas to ensure that required standards and procedures for operating and administering the new target environment are addressed.
- Implement an infrastructure solution that can address your reliability, accessibility, and growth requirements by using cloud-native approaches and best-of-breed languages and frameworks. The components of a modernized application have these characteristics:
  - Packaged as lightweight containers
  - Designed as loosely coupled microservices
  - Centered around APIs for interaction and collaboration
  - Architected with a clean separation of stateless and stateful services
  - Isolated from server and operating system dependencies

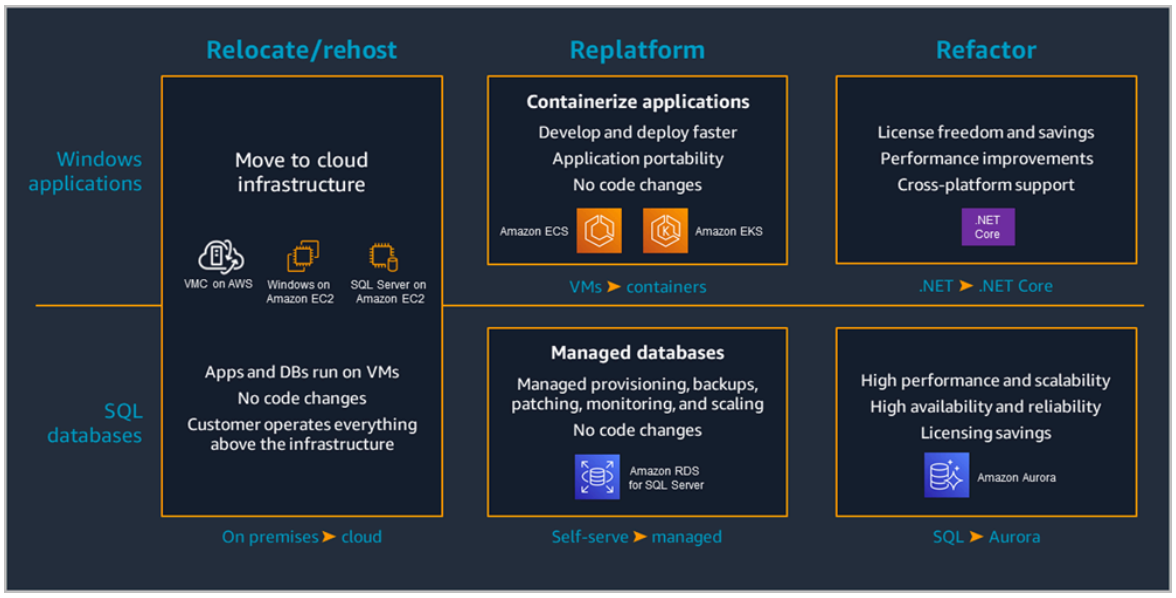
- Deployed on self-service, elastic, cloud infrastructure
- Managed through agile DevOps processes
- Include automated capabilities
- Provide defined, policy-driven resource allocation

Outcomes

- Target state data model design
- Organizational readiness built though training and tool improvements (change management and operational model)
- Regular cadence established for change activities
- Refined operating model and measurement of delivery effectiveness
- Key business case metrics, which are tracked and reported for value delivered
- Continuation of refinement and automation activities
- A modernization roadmap that defines the strategy that is applied to each application and how it can scale
- Preparation and implementation of modernization, including iterative testing deliveries that are synchronized with the new application roadmap

Example

The following diagram shows modernization options for legacy Windows applications.



How-to guide

- Phased approach to modernizing applications in the AWS Cloud

Manage

Relearning efforts are embedded in all modernization activities, to give you a detailed understanding of application characteristics and to reduce any risks that might be caused by subsequent modernization

efforts. Application workloads still need to be able to exploit platform services so that application teams can understand and optimize the runtime characteristics of their application workloads. This means that application teams should treat the operational features of modernized applications like all other application features, and microservice operations effectively become part of engineering. Embracing this DevOps culture in cloud-native operations, as part of building a site reliability engineering (SRE) capability in the organization, is essential to successful modernization adoption. The management phase includes all the elements of effective change management, program management, quality assurance, and service excellence.

How-to guide

- Modernizing operations in the AWS Cloud

Next steps

Before you begin the process of modernizing your applications, perform a readiness assessment to determine whether your organization is prepared to begin the modernization journey. Create alignment between teams so they can adopt the new ways of working. Most companies have a large footprint of legacy applications that generate the most revenue and provide core business capabilities. The stickiness of the architecture, infrastructure, and technology of these legacy applications create complexity and speed bumps for modernization, and further limit the ability of teams to innovate and transform their applications.

The approach to modernization should be incremental but continuous. By following this phased approach, you not only reduce technical debt but help your teams realize cloud benefits faster by scaling seamlessly with self-healing systems, maximize return on investment (ROI) by using cloud-native patterns, offload undifferentiated workloads and dependencies, and improve application performance. For more information about how you can change your operating model and align with a framework that starts with a detailed assessment of current applications, and then modernize those applications by using cloud-native services that can deploy, scale at speed, and manage with efficiency, see the modernization guides in the Related guides (p. 11) section.

Resources

Related guides

- Evaluating modernization readiness for applications in the AWS Cloud
- Phased approach to modernizing applications in the AWS Cloud
- Modernizing operations in the AWS Cloud
- Decomposing monoliths into microservices
- Integrating microservices by using AWS serverless services
- Enabling data persistence in microservices
- Prescriptive guidance for migrating to the AWS Cloud

AWS resources

- AWS documentation
- AWS general reference
- AWS glossary

Document history

The following table describes significant changes to this document. If you want to be notified about future updates, you can subscribe to an RSS feed.

Change	Description	Date
Initial publication (p. 12)	—	December 18, 2020

AWS Prescriptive Guidance glossary

The following are commonly used terms in strategies, guides, and patterns provided by AWS Prescriptive Guidance. To suggest entries, please use the **Provide feedback** link at the end of the glossary.

Modernization terms

business capability

: What a business does to generate value (for example, sales, customer service, or marketing). Microservices architectures and development decisions can be driven by business capabilities. For more information, see the Organized around business capabilities section of the Running containerized microservices on AWS whitepaper.

domain-driven design

: An approach to developing a complex software system by connecting its components to evolving domains, or core business goals, that each component serves. This concept was introduced by Eric Evans in his book, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Boston: Addison-Wesley Professional, 2003). For information about how you can use domain-driven design with the strangler fig pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

microservice

: A small, independent service that communicates over well-defined APIs and is typically owned by small, self-contained teams. For example, an insurance system might include microservices that map to business capabilities, such as sales or marketing, or subdomains, such as purchasing, claims, or analytics. The benefits of microservices include agility, flexible scaling, easy deployment, reusable code, and resilience. For more information, see Integrating microservices by using AWS serverless services.

microservices architecture

: An approach to building an application with independent components that run each application process as a microservice. These microservices communicate through a well-defined interface by using lightweight APIs. Each microservice in this architecture can be updated, deployed, and scaled to meet demand for specific functions of an application. For more information, see Implementing microservices on AWS.

modernization

: Transforming an outdated (legacy or monolithic) application and its infrastructure into an agile, elastic, and highly available system in the cloud to reduce costs, gain efficiencies, and take advantage of innovations. For more information, see Strategy for modernizing applications in the AWS Cloud.

modernization readiness assessment

: An evaluation that helps determine the modernization readiness of an organization's applications; identifies benefits, risks, and dependencies; and determines how well the organization can support the future state of those applications. The outcome of the assessment is a blueprint of the target architecture, a roadmap that details development phases and milestones for the modernization process, and an action plan for addressing identified gaps. For more information, see Evaluating modernization readiness for applications in the AWS Cloud.

monolithic applications (monoliths)

: Applications that run as a single service with tightly coupled processes. Monolithic applications have several drawbacks. If one application feature experiences a spike in demand, the entire architecture must be scaled. Adding or improving a monolithic application's features also becomes more complex when the code base grows. To address these issues, you can use a microservices architecture. For more information, see Decomposing monoliths into microservices.

polyglot persistence

: Independently choosing a microservice's data storage technology based on data access patterns and other requirements. If your microservices have the same data storage technology, they can encounter implementation challenges or experience poor performance. Microservices are more easily implemented and achieve better performance and scalability if they use the data store best adapted to their requirements. For more information, see Enabling data persistence in microservices.

split-and-seed model

: A pattern for scaling and accelerating modernization projects. As new features and product releases are defined, the core team splits up to create new product teams. This helps scale your organization's capabilities and services, improves developer productivity, and supports rapid innovation. For more information, see Phased approach to modernizing applications in the AWS Cloud.

strangler fig pattern

: An approach to modernizing monolithic systems by incrementally rewriting and replacing system functionality until the legacy system can be decommissioned. This pattern uses the analogy of a fig vine that grows into an established tree and eventually overcomes and replaces its host. The pattern was introduced by Martin Fowler as a way to manage risk when rewriting monolithic systems. For an example of how to apply this pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

two-pizza team

: A small DevOps team that you can feed with two pizzas. A two-pizza team size ensures the best possible opportunity for collaboration in software development. For more information, see the Two-pizza team section of the Introduction to DevOps on AWS whitepaper.