Holistic Framework for Cloud Native Computing and Architecture

Innovative Development Cloud Native Framework

By Jeffrey A. Fisher, Cody Mortimer, Trevor Kirchner, and Marc Portner

Contents

Executive Summary

Despite cloud computing promises to deliver lower IT costs, improved reliability, scalability, high performance, and other benefits, many organizations have struggled to achieve these goals through cloud architecture. The technology to drive innovation while lowering cost 'can be realized' through full adoption of the cloud providers native architecture. Organizations embracing cloud native have found not only cost savings but agility and flexibility from automation and aligned resource management. A clear roadmap and guidance enable organizations to adopt cloud native architecture with clearly defined outcomes. This paper seeks to address this issue by providing a comprehensive overview of cloud native architecture, including its key components and best practices for adoption. By following these guidelines, organizations can achieve a higher return on investment with cloud rich capabilities, leading to increased competitiveness and success.

Introduction To Cloud Computing

Cloud computing takes the best practices from IT history and builds a low touch infrastructure to optimize IT technology solutions for business. These practices build the foundation for the creation of the next generation of IT software and infrastructure corporate assets. After adopting cloud computing, IT and the business are one team fulfilling a mission to create collaborative non-siloed software assets which are agile, lean, and resilient with frequent releases.

IT project teams utilizing cloud computing have more advanced tools and can achieve more in less time, while also creating applications that are visually appealing and intuitive. The supporting cloud infrastructure is set up as code to be deployed or destroyed from either direct request or triggers from telemetry monitoring. Subsets of applications are deployed through automation via repeatable means, instead of larger, less frequent, multi-function releases from a monolithic stack. IT project teams adopting cloud computing concepts can move from ideation to feature release within weeks.

To build a cloud native architecture, a company will build assets on a cloud computing infrastructure. The cloud computing infrastructure supports serverless architecture, backend services, telemetry, security, and the application as a set of microservices. Microservices are granular applications (APIs) which employ an event driven architecture. Microservices are developed, deployed, and released via containers, and are secured and managed through DevOPS/DevSecOps practices. All the efforts are delivered, improved upon, and perfected by a DevOPS/DevSecOps process and managed via Agile management.

To achieve a truly cloud native architecture, teams must prioritize the most cost-effective means of deploying corporate assets to meet current and future organizational needs. This includes determining how to maintain, operate, and release applications. This requires collaboration between product owners and system owners to ensure alignment with corporate goals. This collaboration of product owner and system owner aligns corporate needs by adopting lean practices and striving for maximum efficiency. This enables teams to build systems that operate at the lowest possible cost.

Summary of Cloud Computing

What is it?	Why do it?	How to measure success?
Cloud Computing provides on-demand delivery of configurable computing resources over the internet. Resources include networks, servers, applications (yours, COTS), storage, tools, and other IT resources. Cloud computing reduces major hardware investments by taking advantage of cloud scalability for a lower pay-as-you-go cost, as well as shifting training, maintenance cost, and responsibilities to the cloud service provider.	One of the most beneficial reasons to switch to cloud is the IT cost savings. Another is eliminating on-premises risks associated with hosting data. Also, cloud computing enables applications to run with basic computing setup, but scale to meet peak times (scalability). Cloud providers provide best of breed security practices and services, enabling organizations security group to secure applications.	Key Performance Indicator (KPI) examples include operational computing cost, uptime, scalability, and performance. Additionally, cloud providers offer tools to monitor and/or manage applications to optimize performance.

Migrating an organization and its applications to the cloud is a complex and carefully planned process. To do so, organizations must design a private cloud instance with internet gateways, subnets, and identity access management practices that allow current instances to run seamlessly during the migration and validation process. In addition to the "lift and shift" approach, teams may need to tinker with and adapt the application for scalability, containerization, and the use of new cloud provider tools to better understand how the application runs and identify points of failure. By doing so, they can overcome global latency issues and unlock the full potential of the cloud. Additionally, cloud providers offer services that enable organizations to enhance their enterprise knowledge and optimize their operations, while paying for only what they use.

Capability Loss from Being Cloud Enabled

Failing to adopt a cloud native architecture can result in missed opportunities to enhance performance, scalability, resilience, operational efficiency, security, user satisfaction, team abilities, and monitoring capabilities. Oftentimes, teams migrate applications to the cloud only to discover that one or more of these factors is not properly aligned. This can result in lost opportunities, including those outlined below. By adopting a cloud native architecture, organizations can address these issues and unlock the full potential of cloud computing.

While lifting and shifting an application to the cloud can provide numerous benefits, not every cloud offering will be suitable for every application. For instance, an application may require named IP addresses between servers as part of its load balancing design, which can hinder scalability. In such cases, the application may need to be re-architected or refactored to take full advantage of the cloud's capabilities. Operational efficiency is another area that can benefit from cloud adoption. Application teams can use Infrastructure as Code (IaC) templates to streamline deployment and management processes. However, Mean Time to Resolve (MTTR) may remain the same initially, as the monolith application has just

transitioned to a new infrastructure. Organizations will need to gather insights from logs before optimizing the MTTR. Event logging for the application remains relatively the same.

In summary, moving an application to the cloud creates opportunities to modernize and optimize the application, but this does not happen automatically with a lift and shift or lift, tinker, and shift approach. Organizations may need to re-architect, refactor, and implement new practices to fully leverage the benefits of cloud computing.

Control Plane and Data Plane

Cloud computing architecture is divided into two planes: the control plane and the data plane. The control plane is responsible for managing the underlying infrastructure and handling administrative tasks such as provisioning, scaling, and monitoring. In contrast, the data plane is responsible for storing and processing the actual data.

Having a cloud provider manage the control plane offers several benefits, including:

- Reduced Overhead: The cloud provider handles the underlying infrastructure, freeing up the customer to focus on their applications and data.
- Improved Scalability: The cloud provider can quickly and easily scale resources as needed, which is critical for applications with variable or unpredictable workloads.
- Increased Reliability: With the cloud provider managing the control plane, the customer can benefit from the provider's expertise and investment in maintaining a highly available and reliable infrastructure.

Conversely, organizations may choose to manage the data plane themselves. This entails overseeing their cloud network, all its components, the setup of applications, data, 3rd party components, and related security. By doing so, they retain control over their application and data. The organization ensures that their business operates with all the data being stored, processed, and managed according to their own policies and standards.

Implementing a cloud architecture that leverages the strengths of both the control plane and the data plane can result in significant cost savings. For example, customers can benefit from the cost savings associated with utilizing the cloud provider's infrastructure by storing and processing their data in the cloud, while still maintaining control over the data plane to ensure that it meets their specific requirements. Additionally, organizations can design their architecture to take advantage of the scalability and reliability offered by the cloud provider. This allows them to easily add or remove resources as needed, ensuring that their applications are always available and performant.

Cloud Native Computing and Architecture

The value proposition of cloud native computing architecture involves a long-time debate in IT, do we build it or buy it. In his book A Seat at the Table: IT Leadership in the Age of Agility, Mark Schwartz states, “Today’s choice is no longer really between build and buy. It is between quickly assembling best-practice frameworks with continuous user feedback and then continuing to adapt the system over time as the business changes versus buying an undefined stream of future services from a vendor who doesn’t know your business and doesn’t have financial incentives to support you.”

Moving VMs to the cloud starts a process, but architecturally the application remains constrained as a monolithic stack. Changing architectural practices continues to be where most companies struggle. Adopting Agile and DevOps to support the organizational practices can be put in place within 6-9 months. However, shaking off the old design habits, and instead thinking cloud native, means implementing small features and tearing down decades of full application modeling and implementation. This requires new ways of thinking about design, implementation, delivery, managing, and operations.

Application teams work towards organizational goals within a practice of selection of the highest return value for the corporation. They focus on those application systems which have the greatest potential to improve the bottom line. Teams will be created and dissolved accordingly. Some applications may have a current team as the application is deemed complete and perfect for what the budget allows. Product owners may have budget to build a microservice or sets of microservices to create a profit stream or to minimize the cost of modernizing existing systems. Examples of microservices are simple validation routines, payment routines or the automation of an order fulfillment process.

To achieve the desired cloud and architecture goals may require reorganizing the way people think about the process of architecture design and how the organization IT functions support the business processes. Changing how individuals think about, execute, and implement IT processes takes careful planning and training, but it is a necessary step to achieving full cloud adoption. The migration to cloud hosted services and removal of existing on premise physical architecture leads to significant saving according to Nick Galov, who states, "The average savings from cloud migration comes to around 15% on all IT spending. Small and medium businesses benefit the most, as they spend 36% less money on IT that way."

Benefits of a Serverless Architecture

Four key features of going to a cloud native architecture are scalability, flexibility, reliability, and ability to lower operational cost.

- Improved Scalability: Serverless architectures automatically scale resources as needed, allowing the application to handle spikes in demand without any manual intervention.
- Increased Flexibility: Serverless architectures allow developers to focus on writing code, rather than managing servers and infrastructure. This leads to faster development times and increased flexibility in terms of the types of applications that can be built.
- Increased Reliability: Serverless architectures are typically built on top of highly available and scalable infrastructure managed by the cloud provider, which results in improved reliability and uptime for the application.
- Lower Costs: With serverless architecture, customers only pay for the specific functions or operations that they use, rather than for an entire server or virtual machine that may be mostly idle. This results in significant cost savings, especially for applications with variable or unpredictable workloads.

Cloud Native Computing and Architecture

What is it?	Why do it?	How to measure success
"Cloud-native technologies empower organizations to build and run **scalable applications** in modern, **dynamic environments** such as public, private, and hybrid clouds. **Containers, service meshes, microservices, immutable infrastructure, and declarative APIs** exemplify this approach. These techniques enable loosely coupled systems that are **resilient, manageable, and observable**. Combined with **robust automation**, they allow engineers to make high-impact changes frequently and predictably with **minimal toil."** **Cloud Native Computing Foundation**	Companies will achieve lower cost of infrastructure by scaling on demand. Full adoption leads to tighter, better run teams who can oversee operations, support, and development. Increase competitive advantage by accelerating time to market, reduce code base complexity, increase interoperability, gain development, and release independence, and provide resiliency and auto-scaling through microservices and container orchestration.	Multiple factors should be measurable in being cloud native: • Lower operational cost. • Improved application service availability. • Reduced Mean Time to Correction. • Lower technical debt. • Better trained teams to implement change. • Improved speed to market • Reduced duplicative or unnecessary efforts.

Adopting cloud native technologies offers numerous benefits to organizations, including reduced operational costs, improved application service availability, reduced overhead, improved system performance (scalability, reliability, flexibility) lower mean time to correction, lower technical debt, teams aligned with business objectives, improved speed to market, and reduced duplicative or unnecessary efforts.

Cloud computing infrastructure is supplied by providers like Google, Microsoft, AWS, IBM, or Oracle. The infrastructure runs in a serverless environment. Building applications as microservices on the cloud provider's platform allows organizations to fully utilize the latest technologies. Microservices are built in containers and are isolated and secured within the infrastructure platform.

Defining a cloud network and infrastructure requirements as code enables organizations to modernize deployments through automation which creates consistency, enables scalability, provides version control and implements security best practices. Infrastructure as Code provides cost savings in the deployment of code and by enabling increased scalability.

Scalability is achieved by increasing the platform's capacity to manage demand for usage. Kubernetes or a similar technology defines a container image as a ready-to-run software package (with all the required components to run an application). Service-to-service communications and internal application concerns are handled by a service mesh, while declarative APIs keep objects in sync with a central code source.

Observability is an important aspect of cloud native computing and is achieved through telemetry. It enables organizations to monitor and understand the behavior of their applications and infrastructure, and make informed decisions based on the data collected.

Cloud native computing offers organizations the ability to build and run scalable applications in modern, dynamic environments, resulting in reduced costs, improved service availability, and increased competitiveness. Full adoption of cloud native technologies enables organizations to optimize their use of the latest technologies and make the most of their cloud infrastructure.

Microservices

What is it?	Why do it?	How to measure success
Microservices are a key component of cloud native architecture. A cloud native microservice is a small, independent, and self-contained unit of software that performs a specific function and communicates with other microservices through APIs.	Scalability, resilience, faster time to market and improved maintainability.	• Performance response time and reliability of the microservice. • Scalability to meet changing demand as necessary. • Resilience of being able to continue operating, even if other parts of the system fail. • Operational efficiency occurs in the cost and effort required to deploy, manage, and maintain. • Security protects sensitive data and prevents unauthorized access. • Business value of the microservice to overall business goals.

Again, utilizing microservices in cloud native architecture provides several benefits: scalability, resilience, faster time to market and improved maintainability.

For scalability, microservices can be scaled independently, allowing organizations to respond to changing demand and user needs. Decoupled microservices can be scaled verses an entire application being scaled. In this scenario, cost factors will be different, as the decentralized architecture enables scaling to occur independently requiring less computing resources and thereby reducing the company spend. Secondly, if system failures occur, microservices can continue to operate, increasing the overall reliability and availability of the system.

A microservice provides a small unique service, say order management handling the creation, processing, and fulfillment of customer orders. If each subcomponent is a microservice itself and the business wants to make changes to the workflow for how orders are processed, a change to just a microservice component occurs and not the entire order management system. This reduces time to market for business and system process improvements. This faster time to market includes bug remediation or the MTTR, which can be achieved from a DevOps/DevSecOps model. The maintainability of microservices

improves as they are developed, tested, and deployed independently, and thus become easier to maintain and update.

Cloud native microservices can include both backend services, such as APIs that provide data or perform specific functions, as well as user interfaces, such as web or mobile apps that provide a way for users to interact with the system. The decision of whether to include a user interface will depend on the specific requirements of the microservice and the overall architecture of the system.

In general, a microservice should be small enough to be easy to understand and maintain, but not so small that it becomes difficult to deploy and manage. The goal is to strike a balance between service responsibility, cohesion, coupling, scalability, deployment, roadmap, evolution, and team size. All these factors will adjust based on what is appropriate for the specific requirements of the microservice and the overall architecture of the system.

Changing a monolithic application system into microservices can be a complex and challenging process. It's important to keep in mind that this process can take several months or even years, depending on the size and complexity of the monolithic application. It's also important to involve key stakeholders and development teams throughout the process to ensure that everyone is aligned with the goals and outcomes of the transition. IT directors, stakeholders, managers, and IT architects must first understand the implications and benefits of microservices. They must also embrace cloud native architecture and all its components as part of a multi-year cloud computing transition plan. The plan will include the alignment of both people and financial resources. Adopting microservices is a change in architecting, planning and execution regardless of the organization's current experience and usage of Agile and DevOps methodologies. Once all the plans have been presented to senior management, implementation of efforts occurs based on funding and on ROI.

Manage Cloud Data / Backend Services

What is it?	Why do it?	How to measure success
Any service on which the application relies for its functionality. Examples include data stores, messaging systems, caching systems, security, or line of business functionality.	Applications require supporting applications and resources. Enables optimization of resources. Enables applications to be highly available, scalable, and departmentalized.	• Code reuse is high, and duplication is low. • Enterprise resources are fully used without one-offs.

All the application's backend services require orchestration with microservices and other components. Backend services include all 3rd party applications, internal APIs, database and other data stores, notification services, queueing systems, caching services, streaming services, monitoring services, tracking and analytics services, and identity services.

For cloud computing, these services are key to the application, but do not need to be cloud native themselves. For example, Vertex, a global tax company, manages the tax a company needs to collect based on the sale of an item. The product owner could start to build this in house, however, the 'Good to

Great' mindset here would stick to core business and align with a vendor. Maintaining tax tables for all the goods that could possibly be sold globally would take a staff to manage and a system to be managed and maintained. Working with a vendor enables organizations to optimize costs and keep IT focused on creating value.

The Cloud Native Computing Foundation's cloud native architecture definition does not imply data store as a main driver.. A well architected database taking advantage of the cloud provider's data stores and archiving tools can be considered cloud native. Properly integrated the database will be taking full advantage of appropriate archiving and data access.

Data storage can be optimized through several means. Refactoring applications to be cloud native and to use cloud-based data storage APIs, yields high returns. Understanding costs of cloud storage typically depends on the amount of data stored, the frequency of data access, and the level of durability and security required. Companies can opt for either pay-as-you-go pricing or reserved capacity pricing based on their storage requirements. Thus ensuring that data is organized in a way that makes it easy to access and manage in the cloud. A successful optimization will include analyzing current data backup and disaster recovery architecture for strategies that take advantage of cloud storage options. There should also be a review of the configured data access controls, security posture, and compliance requirements to meet the needs of the organization. By making these changes, companies can take advantage of the scalability, reliability, and cost-effectiveness of cloud storage, while ensuring that their data is safe and secure.

Service Meshes

Service meshes are a dedicated infrastructure layer for microservice applications in cloud environments. They provide features such as service discovery, load balancing, traffic management, security, and observability for communication between microservices.

Service meshes are important to cloud applications because they help to address the challenges that arise in microservice architecture such as service discovery, reliability, and security. By providing a common infrastructure layer, service meshes simplify communication between microservices and provide a unified way to manage the behavior of these microservices. This leads to improved resiliency, security, and scalability for cloud applications.

Declarative APIs

Declarative APIs are a type of application API that specifies what an application should do, rather than how it should do it. Instead of specifying a sequence of imperative steps to perform a task, declarative APIs define the desired state of the system and allow the underlying system to manage the process of achieving that state.

In cloud native architectures, declarative APIs are used to manage the deployment and configuration of cloud resources. For example, instead of manually configuring and deploying a set of virtual machines, a cloud native architecture might use a declarative API to specify the desired state of the system in the form of a configuration file. The underlying cloud platform would then take care of provisioning the virtual machines, installing necessary software, and configuring network settings to match the desired state defined in the configuration file.

Declarative APIs are particularly useful in cloud native environments because they allow for automation and simplification of complex operations. They provide a way to manage the entire lifecycle of cloud resources in a consistent and reproducible manner. Additionally, declarative APIs enable versioning and tracking of changes, making it easier to roll back to previous configurations if necessary.

Cloud Observability / Telemetry

What is it?	Why do it?	How to measure success
Observability, also known as Telemetry, involves the monitoring, administration, and management of cloud infrastructure. Cloud telemetry uses software tools to record and analyze information about the IT infrastructure that would otherwise be difficult to gather. Telemetry provides insight into patterns which can help organizations lower costs, improve customer experiences, resolve defects, and optimize systems holistically. Cloud monitoring uses automated and manual tools to manage, monitor, and evaluate cloud computing architecture, infrastructure, and services.	To understand the sizing of the infrastructure; to understand your threats and to map out better means to optimize your cloud provider offerings. Additionally, since you pay for cloud services, you will want to use telemetry tools to manage your optimal infrastructure to minimize the cost of service. For cloud management, telemetry is critically important: to the human eye, IT infrastructure looks very similar whether the hardware is performing optimally or not. Telemetry gives IT professionals the ability to observe components and monitor applications in a deeper way, with metrics that track performance, utilization, energy consumption, and more.	• Means to lower cost appear and are realized. • Infrastructure and application errors and anomalies are revealed and corrective action utilized to optimize infrastructure and application configurations. • Advanced techniques facilitate identification, reporting and adjustments to the configuration for application and infrastructure optimization.

Telemetry is an important component of cloud native architecture and is used to monitor and gather data about the performance and behavior of an application. Through application monitoring, telemetry is used to monitor key metrics and performance indicators for each component of the application, providing insight into how the application is functioning in real-time. Telemetry also logs the events and messages generated by the application, providing a historical record of the application's behavior and performance. Performance optimization can identify areas for improvement and allow data-driven decisions about how to optimize the application. Additionally, telemetry is used to diagnose and debug issues with the application, providing valuable data for troubleshooting and fixing problems. Telemetry data is used in capacity planning so that organizations can anticipate demand and scale their resources accordingly. Lastly, telemetry data is used to demonstrate compliance with regulatory and security requirements, providing evidence that the application is operating in accordance with established standards.

Telemetry is typically integrated into the cloud native architecture using tools and platforms specifically designed for monitoring and logging. By incorporating telemetry into the architecture, organizations can gain a deeper understanding of their cloud native application, enabling them to make informed decisions and continuously improve their software delivery processes.

Cloud Security

What is it?	Why do it?	How to measure success
Cloud security is a collection of procedures and technology designed to address external and internal threats to business security. Cloud security refers to the technologies, policies, governance, and services that protect cloud data, applications, and infrastructure from threats.	• Provide a secure computing environment. • Enable secure deployment and management of applications. • Foster collaboration and communication.	• Reduction or elimination of application or infrastructure security incidents. • Application-level passes vulnerability assessments. • Application-level follows compliance standards such as PCI DSS, HIPAA, and SOC2. • Incident response is treated immediately and fully resolves the issue.

Cloud Native Security Architecture is a set of principles, guidelines, and best practices for securing cloud-based applications, services, and infrastructure. The architecture focuses on security from the ground up, which means that security is integrated into the design, development, and deployment of cloud-based systems.

All cloud security attempts to provide a secure computing environment: this includes the protection of data, applications, and infrastructure from cyber threats, data breaches, and unauthorized access.

At an organizational level, the CSO will want to build a security culture where adoption of security best practices is undertaken and general awareness among employees is fostered. Secondly, regularly monitoring and analyzing security threats and trends can help organizations stay ahead of potential attacks and measure the effectiveness of their threat intelligence capabilities. Lastly, regularly assessing and mitigating security risks can help organizations measure the success of their risk management efforts.

Cloud Native Security Architecture refers to the security of the underlying infrastructure, including the physical and virtual components that make up the cloud environment. This includes the secure deployment of cloud infrastructure, network security, and data center security.

An Identity and Access Management (IAM) system manages and maintains the management of user identities and their access to cloud-based resources. IAM includes authentication, authorization, and access control policies. This also applies to user creation and termination.

Application security includes the security of the applications running in the cloud, including the protection of application data, code, and runtime environments.

Data security involves the protection of data stored in the cloud, including data encryption, data backup and recovery, and data archiving.

Compliance involves ensuring that cloud-based systems meet relevant regulatory and industry standards, including data privacy laws and security standards.

Cloud Native Security Architecture is an essential component of any cloud-based system, as it provides a framework for securing cloud-based applications, services, and infrastructure. The architecture should be integrated into the development and deployment process, and should focus on the protection of data, applications, and infrastructure, while also enabling secure deployment and management.

Machine Learning (ML) and Artificial Intelligence (AI)

What is it?	Why do it?	How to measure success
"Machine Learning is the study of computer algorithms that improve automatically through experience. Applications range from data mining programs that discover general rules in large data sets, to information filtering systems that automatically learn users' interests." IEEE "Artificial Intelligence is the automation of cognition"	• Predictive Scaling. • Real-time monitoring. • Auto-healing. • Optimized resource allocation. • Improved security. • Predictive maintenance. • Personalization. • Improved performance.	• Labor savings. • Threat remediation. • Customer satisfaction.

ML and AI can play a critical role in making cloud native architectures more intelligent, efficient, and secure, enabling organizations to deliver better experiences for their users and drive business outcomes. ML and AI can work behind the scenes to provide predictive scaling, optimized resource allocation to run at peak efficiency, real-time/near-time monitoring and alerting for potential issues, improved security within an organization, data control through response to threats, and personalization for a tailored user experience. Based on the application or organizational goals, operational algorithms can be defined.

People, Processes and Management in Being Cloud Native

Moving to cloud native requires changes in how an organization's people manage the infrastructure, the development processes, product owner relationship management, and operational processes. A requestor/supplier model limits the product owner, system's owner, and DevSecOps / DevOps team to request, prioritize and deliver. This model doesn't build a unified culture required to tackle large initiatives. As teams embark on a cloud journey, people will change from monolithic thinking to molecular problem solving and understand all the subtle changes required to operate IT systems efficiently and effectively. They will also learn how to manage product owner's initiatives, internal IT initiatives, and apply improvements based on any relevant system telemetry.

To be cloud native means shifting an organizations mindset to be Agile and adopting Agile and DevOps or DevSecOps managerial principles. Development and operations are one team, responding to requests from System Owners and Product Owners. In DevOps the team has containerized the product. The product goes through various automated tests and the release is automated as well. Feedback comes from various check points, automated testing, linters, vulnerability testing results, the system owner, and the product owner. DevSecOps teams handle the same duties as a DevOps team but also includes security efforts such as firewall rules, ports, IAM, and such.

DevOps / DevSecOps (Cloud Development and Cloud IT Operations)

What is it?	Why do it?	How to measure success
An operational managerial practice that ties operations and development together, and in doing such, breaks down corporate silos and aligns with product owners and business' goals. DevOps ties together Cloud Native Development and operational support with an Agile practice to resolve issues quickly, bring ideas to development and perfect the product.	Allows IT to keep pace with evolving business requirements. Tightens up the development delivery process, reducing deployment down time from environment to environment. Drives application teams to microservices which have little to no technical debt. Allows developers to spend more time developing and less time deploying and testing.	• Faster development and deployment times. • Improved collaboration and the breakdown of silos. • Backlog highlights the microservices roadmap. Business grows by its ability to deliver quickly to the market's needs. • Cost and time savings from automating the delivery process.

Focused teams deliver more, as the entire process centers on key priorities in a sprint and delivering them. DevOps enables teams, via a set of practices, processes, and tools to automate and bring a single team to manage development, security, and operations. Gene Kim, DevOps guru, states in the DevOps Handbook, "It's not the upfront capital that kills you, it's the operations and maintenance on the back end." Product Owners request changes and results in the large monolith released once a quarter with break-fixes in between. Historically IT did not fully embrace the cost of maintaining an application in production. DevOps brings a single team to not only operate and maintain, but also be responsible for all the enhancements and releases. These team members need to be skilled in robust automation to deliver a perfect application.

Continuous Integration and Continuous Deployment (CI/CD) mean that each day the team seeks to improve the application, not just from an end user usability perspective, but also via telemetry, and changing market needs based on the product owner's request. Through CI/CD, organizations automate internal processes to create, test, deliver and release software of value.

The DevOps culture shift requires the team to see the full view of the application as a corporate asset without technical debt. The team has built the building blocks to release the next version once the effort is determined. This enables teams to quickly absorb requirements, implement, and add value within weeks.

Continuous Integration involves automation of builds, testing and faster time to market (CD) from which improved collaboration, quality, and efficiency (removal of manual tasks) transpire. Also, CI systems provide better visibility into the state of code changes, including the status of builds, tests, and deployments, helping teams to quickly identify and resolve issues.

In a cloud native architecture, CI can be used in combination with cloud services, such as container orchestration platforms, to further improve the automation and efficiency of the software development process.

Continuous Deployment (CD) amounts to the delivery aspect of a release. IaC, as defined earlier, enables organizations to automate the delivery of the infrastructure and the application to a release. Packaging the infrastructure, network, microservices, database objects, components and other objects involved as code to be deployed, enables the organization to create a repeatable delivery process. With skilled individuals who understand programming, automation, and cloud infrastructure, the production and development environments and all in between will mirror each other perfectly.

DevSecOps and DevOps imperatives increase the speed of releases by automating releases. CD also improves reliability by reducing manual errors and improves the reliability of releases, which reduces the risk of downtime and other issues. Proper usage of containers and release automation will ensure each environment contains the next release once all testing has completed successful. Automating the release process via CD also creates credibility in the IT process. Thus, CD systems provide a centralized repository for code changes, making it easier for development and operations teams to collaborate and share code changes. Automation through CD increases efficiency enabling teams to focus on business requirements. CD enables organizations to easily scale their applications and services, helping them to respond to changing demand and user needs. Also, CD systems can include security checks and automated security testing, helping organizations to identify and remediate potential security vulnerabilities before they reach production. In a cloud native architecture, CD can be used in combination with cloud services, such as container orchestration platforms, to further improve the automation and efficiency of the deployment process.

Building out a Continuous Integration / Continuous Deployment (CI/CD) pipeline depends on a multitude of factors such as: infrastructure as code, containers management, version control, application performance monitoring, deployment and server management, configuration management, deployment configuration, software test automation, artifact management, codeless test automation, bug/requirements tracking, and standard published documentation. Quality engineering of CI/CD pipeline will result in end-to-end visibility. Full visibility reduces cycle time of handoffs and creates a consistent workflow.

The tools, best practices and corporate culture create a means to continuously deliver releases, as required once the team shifts to automation.

DevSecOps includes security practices with the infrastructure design, identity management practices, and development practices. As mentioned previously, some advanced teams can manage DevSecOps and others just DevOps depending on the IT organizational structure. However, an IT goal would be to increase teams' knowledge in security practices and identity management and to federate security into the DevOps

team. In many organizations, people grow into specialties, network engineering, development, and security, for example. Cloud computing clearly ties this into a nearly single practice which becomes a DevSecOps team. However, colleges have only started instructing for DevSecOps over the last 5 years. The goal should be to continually build teams to manage the entire microservice effort including security. This enables the Security team to focus more on perimeter attacks and internal threats.

To expand the duties of a DevOps team into being a DevSecOps team, the team in a cloud native architecture will handle and measure success by several key metrics:

- Time to detect and resolve security incidents: This measures the speed at which security threats are identified and remediated, indicating the effectiveness of the security practices and processes in place.
- Compliance with security standards: The regular monitoring and assessment of security compliance with industry standards such as ISO, SOC, and PCI, helps determine the overall security posture of the organization.
- Vulnerability management: The number of vulnerabilities detected and remediated, as well as the time it takes to do so, shows the effectiveness of the vulnerability management program.
- Security breach rate: The frequency and severity of security breaches indicates the overall security of the systems and processes.
- Feedback from stakeholders: Feedback from development, operations, and security teams, as well as from customers, helps measure the effectiveness of the DevSecOps approach in meeting the needs of all stakeholders.
- Continuous integration and deployment (CI/CD) pipeline security: The regular assessment of the security of the CI/CD pipeline, including the tools and processes used, helps ensure the security of code as it moves through the pipeline.
- Automated security testing: The use of automated security testing tools, such as security static analysis and dynamic testing, helps to identify potential security issues early in the development process.

Agile Management

What is it?	Why do it?	How to measure success
Agile management is a people-focused, results-oriented approach to work that embraces organizational change based on the rapidly changing world and current information. Agile centers around providing value as soon as possible, adaptive planning, self-organization, and short delivery times. It's flexible, fast, and aims for continuous improvements in quality. Focus is on the product and the process of perfecting the product to maximize benefit.	Increase the velocity of delivered effort as product to improve operations and the profitability of the organization. Agile management works to lessen the time of delivery and quality of the product or service for the organization to maximize the benefit of the feature requested by the business.	• Increased team morale from member success. • Ability to meet deadlines and deliver within sprints. • Product owner and business satisfaction. • Effective communication and collaboration between development and business operations, as well as between different departments, are key factors in successful project delivery. • SAFe alignment occurs. • ROI conversations occur frequently between the PO and team in decision making.

In a cloud native architecture, Agile project management provides a flexible and efficient approach to delivering software, enabling organizations to quickly adapt to changing business needs and deliver value to customers. Agile management is about execution. Larry Bossidy and Ram Charan have stated, "execution is the ability to mesh strategy with reality, align people with goals and achieve promised results" (Execution: The Discipline of Getting Things Done).

Agile project management helps organizations to deliver value to customers faster, by breaking down complex projects into smaller, more manageable pieces, deliver incrementally, and therefor enabling faster time to market. Agile project management emphasizes collaboration between development, operations, and business teams, ensuring that everyone aligns with project goals and objectives. Through a flexible framework of management, organizations quickly respond to changing requirements, by incorporating regular feedback and iteration into the development process. This in turn creates better alignment with business goals by prioritizing features and functionality that deliver the most value to customers. Additional emphasis is on the importance of automated testing and continuous integration, helping organizations to deliver high-quality software that is free of bugs and security vulnerabilities. Agile project management helps organizations to scale their technology initiatives quickly and efficiently, allowing them to respond to changing demand and user needs.

Final Thoughts

Embracing a cloud native architecture for an organization requires a change of focus on delivery of infrastructure and software as a single commodity, where features, functions, risk, rewards are coupled with costs and liabilities. Companies will need to change their objectives to focus on a single thread of work, so developers can execute and release it, within days after a sprint review with the product owners. The product owner will oversee multiple microservices into a platform service area. All deliberately trying to match or exceed market trends for desired work products. The DevOps platform for execution will ensure the idea becomes a delivered microservice component via a set of Agile sprints. Delivery is from releases via containers encapsulating both infrastructure and application details. Operations and maintenance are managed by the DevOps team which created or were set up and designed to operate the component set of the application. This is the achievable goal of going to a cloud native architecture.

Through all the changes, companies must optimize their people and financial resources to bring about a cloud native architecture across their organization. A cloud native architecture will create a pay per usage model which enables technology to be a growth engine. The organization will manage the data plane and the cloud provider will manage the control plane. Organizations cannot be happy with just adding components to an application. Organizations must realize that being cloud native can optimize resources which can lead to enhanced market share.