((CENTRIC))

WHAT IS MODERN SOFTWARE DELIVERY?

Software Delivery at Speed and Scale

By Joseph Ours | www.centricconsulting.com

INTRODUCTION

In the business world, software powers nearly everything your company does, no matter the industry. In fact, it has been said that because they need software to deliver key business objectives, every company is essentially a software company – whether they realize it or not. Software helps companies automate operations, increase efficiency, build or maintain their reputation and remain competitive in their market.

In today's competitive business landscape, the rapid development and continuous innovation of software are essential for any company's success. Delivering effective and useful software at scale can significantly impact a company's ability to compete in the market and maintain a strong reputation. Conversely, dismissing the importance of ongoing software innovation can lead to detrimental consequences for both the company and its customers.

There have been numerous highly public and high-profile software failures have caused severe disruptions, costing billions of dollars and tarnishing brand reputations. For example, a software error at London's Heathrow Airport resulted in the cancellation, delay or disruption of over 100 flights. Tesla undervaluing SolarCity Corp during its acquisition and the infamous Y2K bug are further examples of costly software failures that have put companies out of business and resulted in significant financial and reputational damage.

While not all software development failures lead to catastrophic losses, every company is exposed to risks stemming from inadequate software development practices. Such risks include tech-based disruption as competitors develop superior solutions, user experience-based churn as customers switch to higher-performing platforms, and reduced profit margins due to the failure to optimize cost-controlling operations.

INTRODUCTION

The persistent question remains: how can my organization achieve quality software delivery at speed and scale to remain competitive in the marketplace?

Our answer: modern software delivery.

Businesses must keep pace with, and ideally outpace, their competitive markets to remain viable. This is particularly true as today's customers (and employees) are accustomed to highly available applications, constantly functional and consumable anywhere, anytime and on any platform. These forces are causing businesses to constantly reevaluate their strategies to gain, control and manage their markets. Unfortunately, the way most companies deliver software and **technology** is insufficient to meet these demands.

Every organization faces challenges from traditional competitors and new attack vectors to standard innovation and technology disruptors. The ease, availability and ubiquity of technology magnify the frequency and scope of these challenges.

There is a myriad of internal reasons why IT struggles to keep pace:

- Technical debt, or prioritizing speed over functionality that you must later reconfigure, is a drag on velocity and increases overall ownership costs.
- Monolithic systems impede agility by design. These software models are independent of other applications, which makes scaling a challenge as you must update an entire tech stack for every functional update.
- Teams are underdeveloped — they are forced together, resulting in poor productivity.
- Long business-case funding and approval cycles bring innovation to a standstill.

The bottom line: Market demand for technology-based solutions creates an ever-increasing need for leveraging technology, which places increasing burdens on IT's ability to develop and deliver quality technology at speed.

There is a solution. But first, let's look briefly at how we got here.

TABLE OF CONTENTS

In this whitepaper, we reveal how modern software delivery came about, what it looks like today and how it can impact your business with the following key sections:

The Three Waves of Software Development Modernization

Software development modernization didn't happen overnight. It has taken place gradually, based on a need for businesses to solve problems like the scalability of legacy systems, underdeveloped teams, technical debt and other internal and external needs. Agile adoption and development operations (DevOps) have addressed some of these software delivery challenges, but only in isolation.

Early on, some organizations experienced longer-term velocity uplift benefits, such as speed to market, reduction in errors or bugs and increased releases, while others only experienced a shorter time for the first feature to market.

THE FIRST WAVE OF MODERNIZATION

The massive uptake in the adoption of Agile methodologies — which focus on better integrating the business and IT functions — was the first wave of modernization.

In this first wave of software modernization, organizations transformed into collaborative and cross-functional teams, which helped them improve transparency and provide some adaptability. However, teams did little to accelerate overall delivery velocity or how much work the team completed during a specific period. According to the 15th Annual State of Agile Report published in 2021, 30 percent of respondents identified 10 or more challenges faced while adopting Agile, with key challenges such as organizational culture, resistance to change and lack of support and skills among the chief barriers to success.

THE SECOND WAVE OF MODERNIZATION

The second wave of modernization began with the introduction of DevOps, a model that combines tools and practices to deliver applications at high velocity and enables product development, deployment and improvement more quickly than other processes. With DevOps, we saw principles of design thinking, Lean and Flow taking hold in the form of automation – specifically, automation pipelines.

Design Thinking: A human-centered problem-solving methodology, design thinking is a process for understanding users and barriers to implementation and then solving those problems using five set phases: empathize, define, ideate, prototype, test. While the phases may seem linear, they're anything but. With each new iteration of a product a company releases the process loops back to previous steps and forward to next steps fluidly.

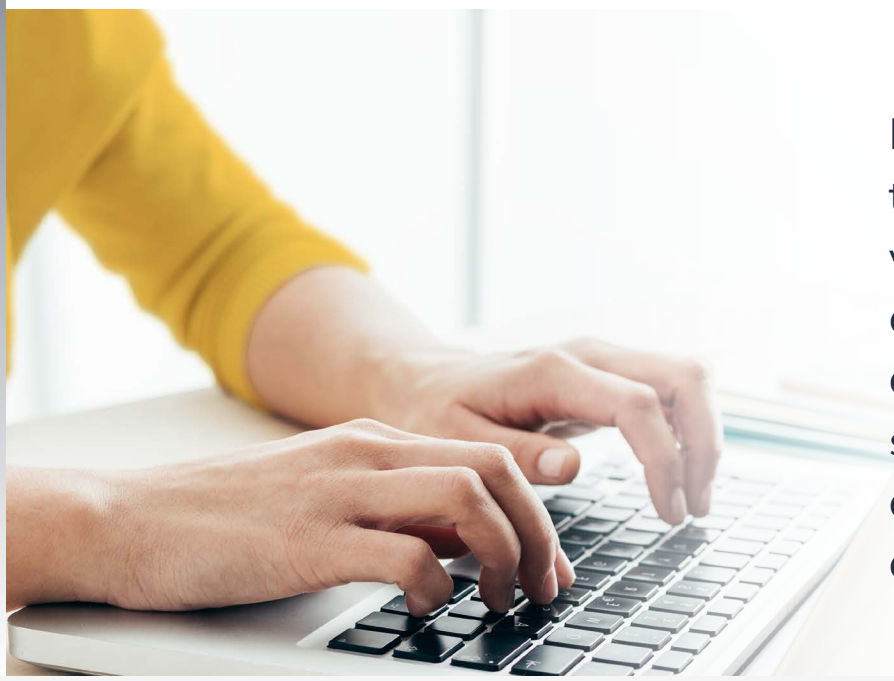
Lean: As the name implies, the Lean software development framework is a set of principles that – when implemented properly – improve efficiency with fewer resources and waste. It centers on customer understanding and ways to provide value most efficiently. Lean is a heavily used model in demanding industries such as software development because it is a streamlined process that focuses on the final product and continuous improvement.

Flow: DevOps process flow is centered around agility and automation with stages of continuous development and improvement. Making operations flow – through waste reduction and process improvement – is Lean's goal.

A DevOps automation pipeline is a set of processes and tools enabling collaboration in software development that speeds up time to market and improves software deployment. Generally, developers have hyper-focused on automating specific segments of the overall delivery process. For example, with a CI build, developers merge code changes into a central repository on an ongoing basis, where they can build and test each revision automatically. The key is that it's automated and repeatable, allowing developers to continually process new code for iterative development, but it runs counter to Lean thinking, in which optimizing Flow is the goal.

Without addressing the ability to introduce change and effectively manage the size of work, there is a practical limit to how much performance automation can deliver on its own. For example, if your team works on stories that take them roughly three days to complete on average, several things happen that we don't often realize. First, this implies a single team member can only execute the "automation machinery," aka DevOps pipelines every three days. This means at a minimum you can never move faster than three days. If the average story size was half a day, then that team member could avail themselves of the pipeline more frequently.

More frequent feedback tends to result in better code over time. However, this is not the only velocity limiter. When work is sized largely, there is significant amount of risk of the unknown. Smaller-sized work has smaller unknown risks. Smaller work translates into smaller risk, and less frequently churn (or re-work), thereby improving overall flow or velocity.



DevOps is more than a tooling platform, it requires vision, redesign of portions of the IT organization, disciplined practices, standardized processes and well-integrated automated tool platforms.

THE THIRD WAVE OF MODERNIZATION

Through leveraging artificial intelligence and data, automation has rapidly sped up the pace of innovation and adoption. Once, automation was a foreign term misunderstood (and even feared) by the outside world. It was an engineer's domain. Today, not only are developers and companies introducing new technology faster than ever, but barriers to adoption have all but disappeared. Even children can navigate a user interface and create AI models, a transformation that took place over about 24 months.

With the **third wave of modernization**, truly effective organizations modernize how they envision, develop, deliver and operate software within the application lifecycle. They transform themselves into collaborative, cross-functional and highly effective automated, self-managed and self-serviceable teams.

Businesses achieve this through adopting modern practices, including value stream mapping or analyzing the current state to develop a future state, design thinking, Lean, cross-organizational DevOps and Lean Agile. Each of these has led to rapid development and deployment of scalable, productive software, reducing time to deploy code from weeks or months to seconds.

The synergy of these approaches represents the third wave of modernization.

Organizations that modeled these proven principles have realized true business value, as seen in improved profitability, productivity and market share. Various State of DevOps Reports have shown many organizations that embody these principles see their market share grow by 50 percent more than their peers.

When software development is the center of your business's strategy, your organization must establish an IT value chain that allows it to stay ahead of the competition by continuously delivering high-quality products at scale and low risk.

As we continue, we will describe how organizations structure their modern software delivery (MSD) platform and further integrate their end-to-end software delivery processes to achieve speed and scale.

A Holistic, End-to-End View of Software (Product) Development

MSD approaches focus on the entire IT value chain, with IT delivery being a system that includes ideation, development, delivery and operations. What IT professionals do in one area greatly affects what they can do in other areas.

Feedback loops are not only necessary for collaboration — they are a natural consequence of any system. Used to validate and solicit feedback during the development process, these communication loops enable teams to address feedback continuously to speed up the development process.

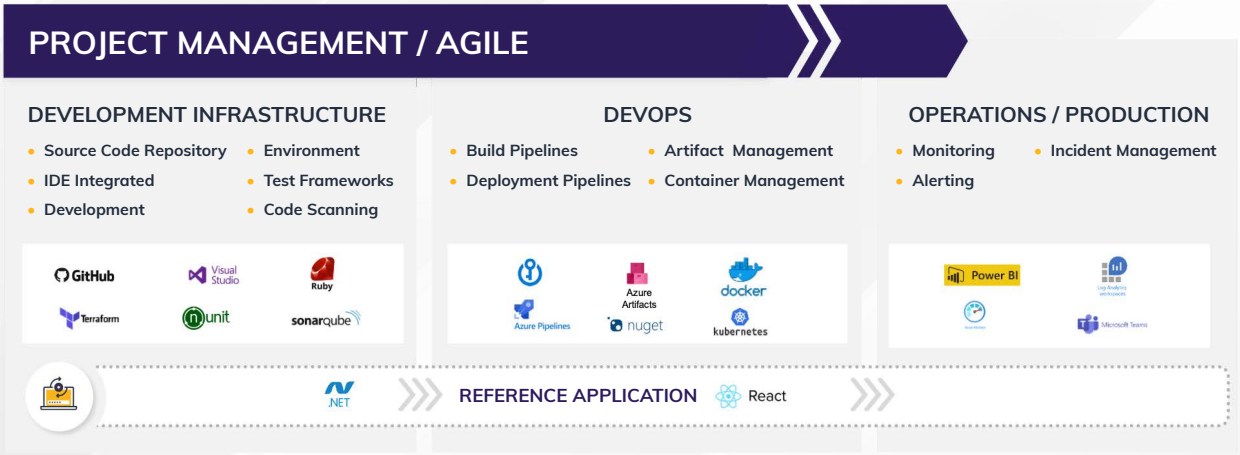


Figure 1: The Modern Software Delivery Pipeline

Figure 1 illustrates the typical components of a modern software delivery organization. All work starts with ideation from the business. **Agile management starts by collecting ideas, vetting them, and converting them into a product vision, roadmap, themes and epics.**

This work leads to development activities, where agile teams break epics down into stories and acceptance criteria. Coupled with modular architecture and infrastructure as code (IaC), this approach of breaking down work into the smallest possible units enables an organization to optimize flow and reduce the risk of changes.

The ultimate goal of MSD is the continual flow of “develop a feature, release a feature” in rapid succession. From this point, DevOps automation takes over in the form of:

- **Continuous Integration (CI):** This DevOps practice is the process of automating the implementation of code changes from several developers into a software program test run.
- **Continuous Delivery (CD):** CI code changes are automatically deployed to a testing environment and released in frequent small batches after testing.
- **Continuous Testing (CT):** Sometimes referred to as continuous quality, CT automates testing at every commit to ensure efficiency.
- **Continuous Deployment:** Continuous Deployment, or continuous delivery, refers to automatic deployment of code changes after completing the above steps. Software pushes the updates directly to end users.

Each phase is responsible for building, validating and packaging software for release, then managing each subsequent feature release. Automation drives overall velocity, but only when features are small and decoupled. Sophisticated infrastructure, security and governance accelerate the process and help achieve optimal velocity and attain quality at scale.

While this flow may initially seem linear, it is a complex interdependence between multiple organizational capabilities, as outlined in Figure 2:

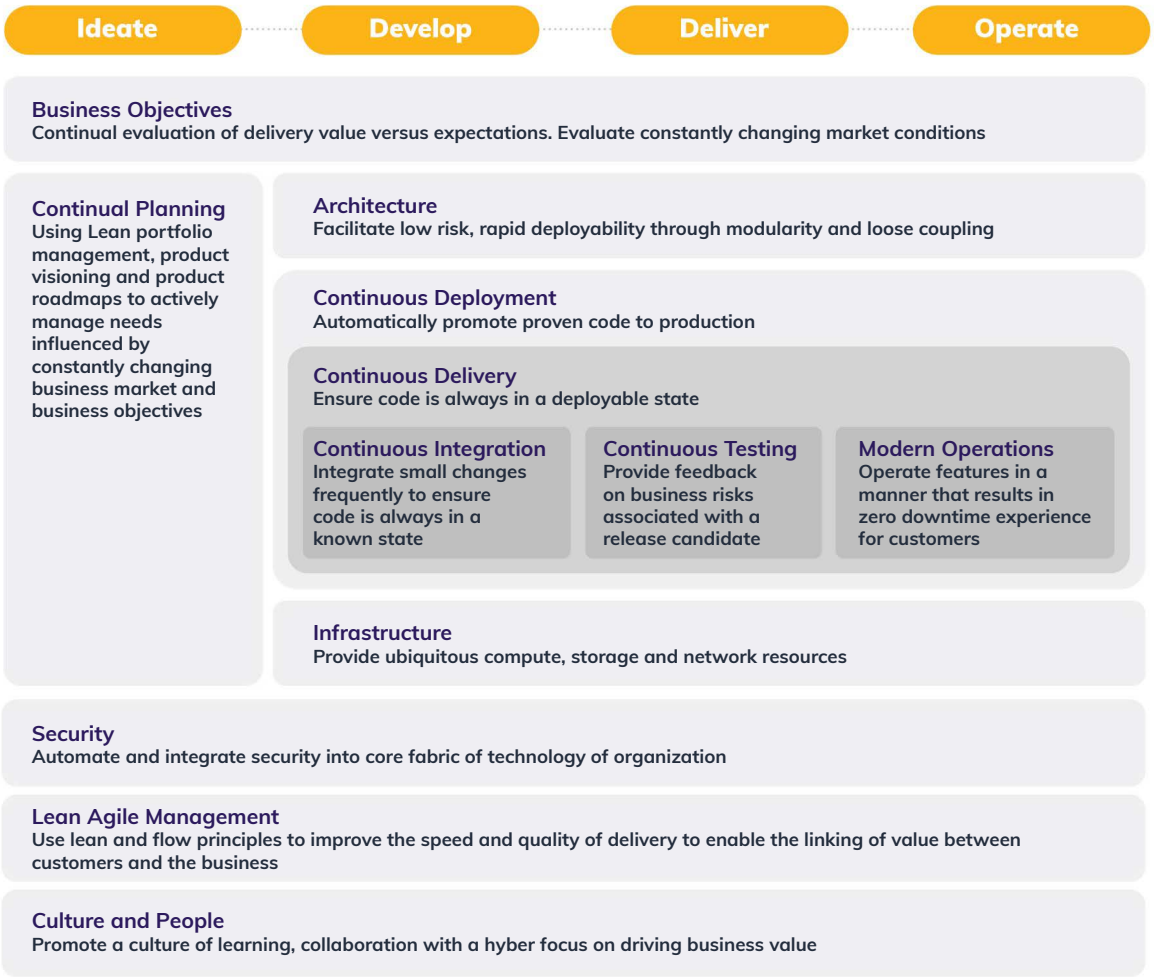


Figure 2: Modern Software Delivery Capability Interlocks

The relationship between several factors regulates the ability to increase speed and optimize flow:

- **The size of each work unit**
- **Transparency in a work unit's current state**
- **Automated handoffs and flows**
- **Automated provisioning**
- **Movement of work through the organization.**

While automation is important, the real work synchronizes multiple types of tasks in an automated flow while reducing risk as you update an existing system. When someone makes a change to software and it breaks, they must quickly remove the offending change to maintain a clean work product. Sophisticated organizations can "unwind" work back to the point of origin – including the associated code commit if needed.

For example, if a developer checks in a piece of code that fails after deploying it to production, a sophisticated system will roll back to the last known working version. The system will completely mark every work stage the piece of code previously passed as a "failure" – even going as far as removing the bad code from the codebase. **When the right quality checks exist at each step, rollback is infrequent.**

Focus on the end-to-end development process, from product ideation to customer value, is critical to establishing a vision for a modern software delivery organization. Developers use feedback loops between creation and operations to help detect potential issues within code during the development process. Ultimately, this enhances the customer experience and helps developers better understand best practices for software development. Closing this feedback loop requires actively monitoring end-user usage and gathering feedback to inform prioritization and product exploration.

Depending on the organization's ambition, this flow of activities can take weeks or mere hours. This is modern software delivery at speed and scale. **The key question: How fast does your organization need to go?** In order to find out, here are some additional questions you should be able to answer:

- **How fast are your competitors developing and releasing features?**
- **Are your developers able to keep pace with CI/CD, and if not, what support do they need to produce quality changes quickly?**
- **What are your customers' expectations?**

As I like to say, "What works for TikTok doesn't work for NASA, and vice versa." The level of investment in a sophisticated system to meet the speed of your business needs and industry will affect your priorities. This is a great opportunity to reach out to an expert, such as a third-party consultant, to help determine what you need and what it will take to get there, including a modern approach to the IT value chain.

How to Tell if Your Company is Hindered by a Traditional IT Value Chain

We generally understand value chain to be the full range of activities a business conducts to bring its product or service to market, from design to delivery. Today, traditional approaches to the value chain can hinder businesses if IT and business operations teams aren't on the same page. **The key symptom that an organization has a legacy approach to development and delivery is the business department's relationship with IT.**

Often, strained relationships between IT and business operations result directly from the business's ever-increasing demands for agility and speed and IT's inability to match the pace expected by non-IT departments. This leads to mismatched expectations and frustration and could even lead to business leaders reaching outside the organization for a solution.

The ubiquity and ease of access to on-demand, targeted solutions make it challenging to steer business offices away — especially with many cloud or software as a solution (SaaS) approaches only a free trial and ongoing subscription away. This enables business leaders to bypass IT departments altogether, creating unmanaged and decentralized IT solutions. In short, it creates shadow IT, or systems deployed outside of IT that address shortcomings of the centralized systems. Without even realizing it, various departments could quickly install a cloud or SaaS app, device or software and introduce a bevy of security risks.

Further, if IT doesn't know about a solution implemented by another department, they can't assess and monitor it for risks, ensure its security or notify parties if there's a duplication of services that will waste company resources.

Failure to achieve predictable delivery within budget is another hallmark of a traditional IT value chain. Even organizations that have transformed into agile practices can struggle to achieve predictable delivery. Often, you will see these organizations have a hyper-focus on velocity and epic burndowns rather than outcome or value. When you see this, you must step back to look at the IT value chain to see how you deliver value (not only features) to the business.

When an organization focuses on the delivery of themes and epics, you typically see them revert to a linear process where they are waiting for value at the end of the process after an accumulation of work has transpired.

With Scrum, this means waiting for value for up to two weeks – due to the nature of accumulating stories in an artificial two-week time box and then releasing them as a batch. Two-week sprints innately shorten the time to achieve value compared to traditional waterfall approaches. Each phase is independent, but many organizations cannot, or will not, release software every two weeks.

The inability or unwillingness to release as soon as teams complete their work means the organization cannot derive value from what they created. This is a form of technical debt – investment in work that cannot create value. While better than the old waterfall environment, this still directly impacts organizations and impedes achieving the full benefits of Agile and DevOps. To achieve maximum value from the value chain, you need a modern software delivery approach.

Better, Faster and More Predictable: A Modern Software Delivery Approach

A modern software delivery approach methodology enables reliable, scalable solutions. But a solution without a clear path to adoption won't benefit businesses or their customers. Teaching teams to adopt an MSD approach empowers them to build and deliver modern software themselves, synergizing Agile management, effective test automation, DevOps practices, modular architecture and cloud platforms, such as Amazon Web Services (AWS) or Microsoft Azure.

By looking at IT as a system and living organism, your teams can address bottlenecks as they arise and ensure targeted investment at current challenges while building toward a future vision.

Figure 3: Application Development

Unfortunately, organizations often target one area at a time without regard to its impact on other areas.

For example, many organizations have undertaken an agile transformation, and some have even implemented a Scaled Agile Framework (SAFe) to scale agile practices. As a result, most teams are operating in Scrum fashion. However, as those same organizations look to implement DevOps, the size of the story directly affects the speed at which they can perform things like continuous integration (CI). After all, you can't integrate or release a story faster than you can develop it.

As organizations mature with Agile and transition to implementing DevOps, they often encounter their next bottleneck: testing. In the 2022 Software Testing Pulsemeter Report, 75 percent of DevOps professionals surveyed said that over 25 percent of productivity is lost to testing, leading to disgruntled developers and costs associated with waiting for bad code and developer turnover.

If you're curious as to why testing is a bottleneck, it is due to a multi-faceted problem starting with the viewpoint of who is responsible for quality. Many organizations shift this burden to a QA or Testing Department. Companies often struggle to include these groups at the right time. In many cases, processes and handoffs are such that QA teams end up testing after development is done, many times after a development sprint is complete. This creates two issues. First, it lengthens feedback loops and introduces churn in the process. Second, teams are overly reliant on manual testing because they don't get enough time to invest in automating tests. Both issues force a bottleneck on the testing process.

A third issue everyone acknowledges at the bar, but seldom speaks about in the workplace, is if the company doesn't consider quality a point of pride of the creator (aka developer), then when push comes to shove, they will sacrifice quality on the altar of time. It is important to develop a holistic approach to actively address bottlenecks when implementing DevOps processes and tooling.

So, while the cloud is DevOps ready, your operations may not be DevOps capable. There are several areas to consider before implementing DevOps:

- **What is your code release and configuration process?**
- **Is your organization secure or agile?**
- **Are developers and teams open to process changes and working cross-functionally?**
- **Are your teams prepared to switch to the continuous development/ improvement model?**
- **How automated are your processes?**
- **Is your organization siloed?**

Only by focusing on MSD as a system can your organization truly achieve the promise of many of these practices and methodologies.

Building an Environment for Quality Across the Value Delivery Chain

Regardless of where you are in the software development life cycle — from defining your product through development, delivery and operations — you must create an environment for quality across the value delivery chain. After all, quality is an attribute of what you build. This simple concept has far-reaching consequences for anybody involved in delivering IT value.

This is not a new concept. Harvard Business School Professor Michael Porter first introduced the value chain in his book: "*The Competitive Advantage: Creating and Sustaining Superior Performance*", and thus, it's now known as Porter's Value Chain Model. And while the business world has changed greatly since he wrote the book in 1985, what hasn't changed is the idea that taking stock of the processes that go into a company's value chain through analysis and then truly understanding how to create value will provide a competitive advantage no matter your company, industry, product or service.

Porter split the value chain into primary and support activities, with several activities that made up each:

- **Primary business activities:** inbound logistics, operations, outbound logistics, marketing and sales, after-sales services
- **Secondary business activities:** procurement, technological development, human resources management and infrastructure.

Through analysis, companies can realize advantages in the form of cost reduction, product differentiation or speed to market. Analysis consists of several steps:

Identify value chain activities: This step involves gathering data, typically using cross-functional teams and project management software. You should map both primary and secondary activities that contribute to the final product.

Determine cost and value of activities: Each process should create value that outweighs the cost of creating value. Simply lowering expenses might be a way to improve value.

Analyze data and determine opportunities for competitive advantage: Interpret data based on your organization's goals and work to understand potential competitive advantages. These may be a cost advantage or a differentiation advantage. If there is a lower-hanging fruit or an activity that will take little effort but produce a greater return on investment, prioritize that improvement.

During the process, it's helpful to also evaluate your competitor's value chains. While an inside look at their operations may not be possible, understanding industry benchmarks and best practices is a good start. This is also an area where an outside consultant can help your company objectively review competitors, as they often have seen many models in action and understand the benefits and challenges industry wide.

To achieve quality at speed and scale, it's important to:

- **Define quality objectives;**
- **Drive quality thinking across the value chain, including primary and secondary activities; and**
- **Analyze and measure quality.**

Because quality is an outcome and not a phase in the process, you must define what you expect, what is acceptable and what is not at each step in the IT value chain. Begin by asking how an activity contributes to improved user or developer experience and how it otherwise creates value.

For instance, is it acceptable to build a product without a defined product vision or roadmap? What if you don't have a defined or expected business outcome? Should you develop stories that don't have acceptance criteria at the story level?

As you drive through the value chain, the specific questions may change, but their nature will remain the same. For example, at the developer/tester level with CI/CD/CT practices, quality is more often an inspection idea, such as "we must test all stories with unit, system and integration tests."

However, what engineering practices are you following to minimize the introduction of coding issues? Are you using linting tools? Code-scanning tools?

You must drive quality throughout the entire chain, from ideation to development to delivery to operations. Failure to drive preventative and inspection activities will always result in churn in your process and drag on overall velocity as you react to issues instead of proactively preventing issues.

Outcomes, Not Activity – Metrics for Success

Agile development teams using DevOps automation pipelines must measure outcomes, not activities. If you don't have a clear picture of what results matter, you can't know if you were successful. Agile development teams need to define what metrics will drive project success and create development processes to support them. You measure success by outcomes, not activities.

To this end, each step of the IT value chain has an associated set of outcome metrics that are valuable to your organization. Examples by phase may include:

- **Build automation:** build-success percentages and build-time service-level agreement (SLA) percentage
- **QA and test suite:** test automation and percentage coverage
- **Deployment pipelines:** deployment downtime, SLA percentages, successful automated release percentages
- **Product management focused on predictable delivery:** lead time or cycle time

Successful metrics for your delivery process will combine leading and lagging measures that focus on what you achieved, not what you did. Remember, no one cares how many nails you use to build a house, but everyone cares that the house is well constructed. Similarly, you must focus on the value delivered to customers, flow and early identification of bottlenecks and constraints to deliver maximum value and realize efficiency.

Take, for example, a private business jet charter company that needed to eliminate its technical debt by modernizing software. With Centric Consulting's help, the company secured $400,000 in AWS funding and migrated more than 100 on-premise applications to Amazon Web Services (AWS) cloud, which led to company agility through the retirement of costly, long-term software licenses.

Another company, a leading supplier of pet food products, personal care products, electronics, and clothing to correctional facilities, implemented MSD practices such as containerization to deliver products more quickly and with higher quality while also increasing customer responsiveness.

A third company, a leading provider of services to the home healthcare and hospice industry, was able to architect and deploy a scalable Windows Virtual Desktop in under three weeks. The MSD solution was cost-effective, secure and enabled employees to access to a remote personal desktop experience on the company's network.

CREATE A COMPREHENSIVE APPROACH TO SOFTWARE DELIVERY

No matter your industry, delivering software at speed and scale is essential in today's business environment. **By adopting an MSD approach, you can combine Lean-Agile thinking, DevOps automation pipelines and cloud-ready platforms with modern software architecture practices.** The result is a holistic, end-to-end IT value chain that delivers software better, faster, and more predictably.

Armed with a modern delivery methodology, you will be better able to adapt and deliver software ready to meet your customers' and employees' needs.

You don't have to try to navigate the modern software delivery journey alone. Centric has a team of experts who can help you every step of the way, ultimately helping you deliver speed at scale while gaining a competitive advantage in your industry.

If you know you need a modern software delivery approach, but you're not sure where to start, contact us for an assessment of your organization's readiness and to learn about the next steps.

ABOUT THE AUTHOR

Joseph Ours is our Modern Software Delivery Practice Lead. He has over two decades of experience in IT and team management. He has a background in crafting strategies for modernizing technology practices for organizations. He is a leader in aligning IT delivery with business value and has a successful record of providing project and portfolio management of large technology initiatives.

Want to learn more about our perspective on agile transformation or Modern Software Delivery?

Let's talk

((CENTRIC))

ABOUT US

Centric Consulting is an international management consulting firm with unmatched expertise in business and digital transformation, hybrid workplace strategy, technology implementation and adoption. The firm has established a reputation for combining the benefits of experience, flexibility and cost efficiency with the goal of creating tailored solutions centered on what's best for your business. Founded in 1999 with a remote workforce, Centric has grown to 1,500 employees and 14 locations across the country and India.

Visit www.centricconsulting.com to learn more.