iVedha

Industrialization of SRE

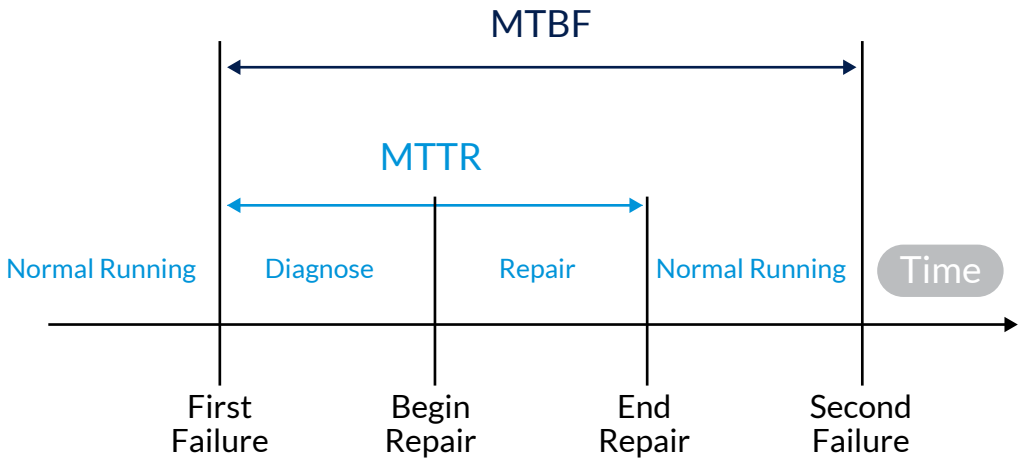
CONTENT

Industrialization of SRE

Mel Conway has said that **"all organizations which design systems are constrained to produce designs which are copies of the communication structures of the organizations."**

There are good reasons why software components end up arranged like human components. Large systems cannot function unless the interfaces between components are carefully defined and change slowly. These complex systems are like contracts that let each side rely on an abstraction that tells them what to expect from the other side. You will notice that technical and people problems are inevitably congruent when a system falls into chronic dysfunction.

Partial Failure Is All Too Common



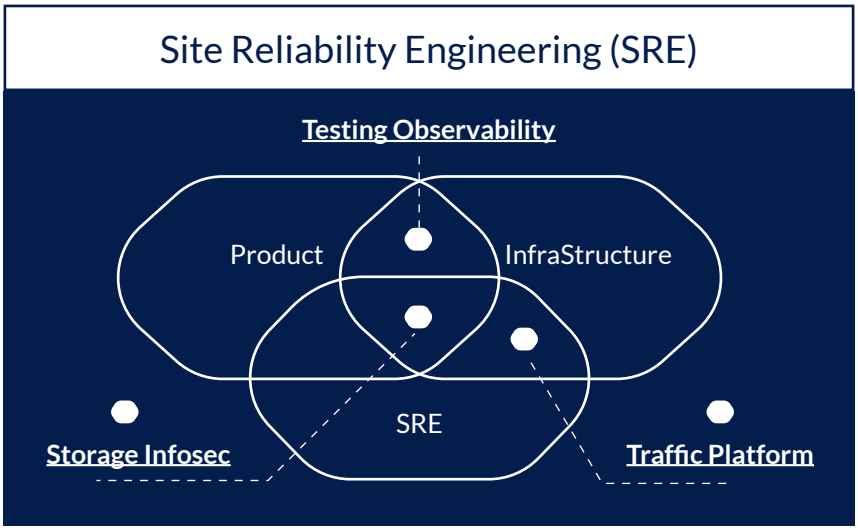
It is commonly known that large systems exist in a state of partial failure at all times. This is an inescapable fact when it comes to the interaction between Mean Time Between Failures (MTBF) and Mean Time to Repair (MTTR).

Operations teams and engineers often undertake big projects to increase MTBF and decrease MTTR. Usually, this is at the level of hardware components because it is the only level at which assumptions of rationality hold well enough for "mean time to anything" to be well understood.

It is always worth the effort for a team to optimize within one accountability domain such as this. But as Edward Betts says in his book, Fault Tolerance, **"When you're looking at the entire system, an increase in fault tolerance beyond 'barely acceptable' tends to be immediately eaten up by another layer."**

There's a growing need among businesses of the 21st century to build scalable systems as they are reliable. Achieving it involves automating various tedious and manual development tasks and eliminating the human error element. While that may not be 100% possible in practical scenarios, the SRE approach comes quite close.

Why Organizations Should Think About Industrializing SRE



Implementing SRE implies that an organization's infrastructure, operations, monitoring, performance, scalability, and reliability factors are accounted for in a nice, lean and automated system. And while this is great, it's not enough.

Culture is a critical aspect driving SRE in sync with business needs. While it is easy to follow pioneering companies, it's impossible to copy their culture and the means to replicate their success. This is especially true when it comes to anti-patterns and traditional remedial baggage. No two organizations have the same infrastructure and business needs. The crux is that we must recognize what is vital for such an initiative to succeed after understanding the fundamentals. Organizations need to define their own success factors after considering their cultural needs. Simply said, strategy and culture need to walk together.

Conceptually, site reliability engineering is cardinal for any software-based organization. It links Ops and Dev in the DevOps. Site reliability engineers formalize the connection that helps link folks together and streamline processes. As the reliability of an organization grows, so does the simplicity across workflows. The bigger your enterprise, the simpler the process must be, but the unfortunate reality is that many organizations increase complexity, reduce transparency, and create wasted effort by adding more individuals. There's a growing need for SRE to be used to understand processes from end to end and emphasize outcomes rather than any particular process stage.

Industrializing site reliability engineering involves bridging the gaps between platform design, development, and operational execution by providing new perspectives on system reliability. When enterprises link software and systems engineering mindsets with operational engineering, they can streamline systems focused on business outcomes, not ticket output. Organizations gain a larger view of the platform, reduce the effort required to support the systems, and spend more time automating and innovating. They evolve towards self-healing and autoscaling systems instead of manual, reactionary troubleshooting.

What Is SRE and Why Should You Bother?

Google is widely recognized as a leader in the IT industry. They have developed a highly efficient and scalable IT infrastructure and have exemplary cloud computing systems. Google's internal management platform handles billions of applications and runs on millions of servers. All in all, its data centers achieve full life cycle management of applications. But even with all that, its most essential innovation in IT remains the field of Site Reliability Engineering (SRE).

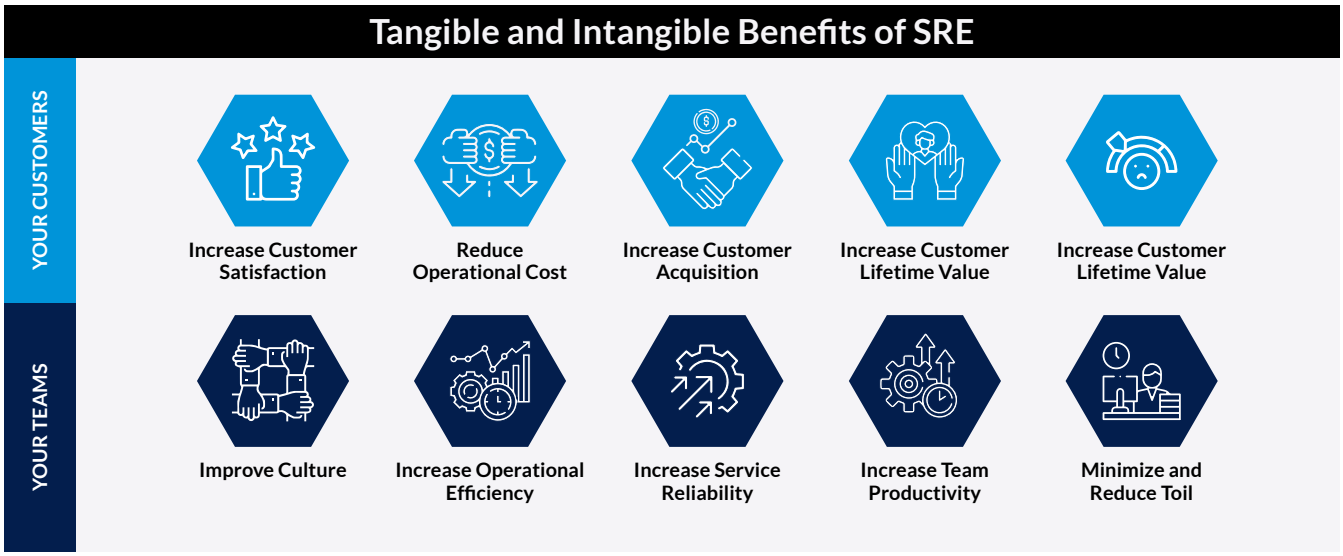
Come to think of it, SRE is more of a non-negotiable and essential practice of delivering the best customer service and reusable automation code. It covers everything from development to operations and maintenance of the entire infrastructure. Apart from that, Site Reliability Engineers are also responsible for managing IT personnel and assisting them to deliver their best work.

Simply said, SRE is a functional, software development-oriented solution to IT operations problems. An SRE's focus on building and monitoring production elements is the reason why any competitive organization's service remains resilient.

Industrialization of SRE is Key to Modern DevOps

The industrial revolution was made up of several phases. At each stage, there was a paradigm shift in the way people and technology worked together to solve complex problems. The industrialization of site reliability engineering is the same — it's a positive fundamental shift in an organization's governance structure that results in better processes followed by superior tooling.

The organizations that adopt open-source technologies quickly are steadily increasing their



competitive edge over others who are undecided or don't intend to innovate. Given that, the industrialization of SRE is more of a structural change rather than a technological one as it aims to make SRE increasingly accessible to any organization looking to streamline its software delivery and make it more reliable.

While agility and innovation are paramount when it comes to software delivery, customer satisfaction can't be put on the back burner. Companies, hence, must avoid making risky infrastructural changes that endanger customer experience.

How Businesses Can Industrialize SRE

The ultimate goal of SRE is to make systems more reliable, thus ensuring the end-users are satisfied with the product. Site reliability engineers spend part of their time on development tasks and the rest of their time is involved in responding to incidents or escalating customer issues. This solidifies that industrializing SRE is a leap toward perfecting the DevOps implementation.

Software products are designed to deliver value to a set of services that customers depend on. As dependency grows, so does the need for SRE. To provide application performance and reliability, development and operations teams must focus on their metrics of success. While the former focuses on the speed of release, the latter must concentrate on maintaining reliability. It is necessary to have a shared responsibility approach to clarify which security tasks are handled by which teams.

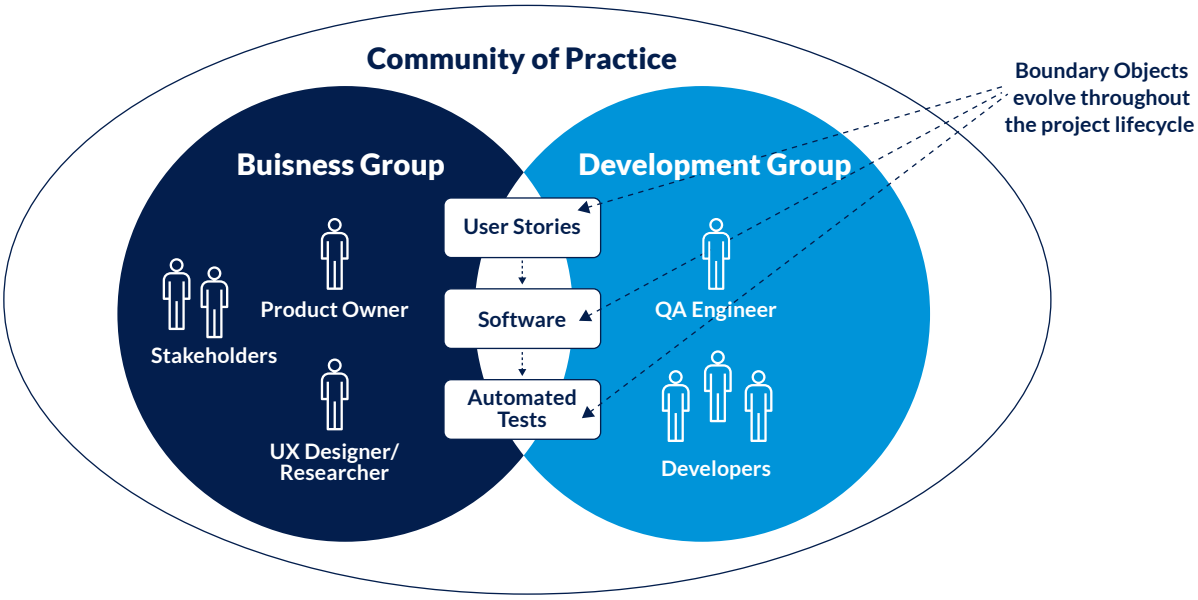
Breaking Silos, Bridging People, Processes, and Culture Together

The first step first towards the industrialization of SRE is to break down and reduce organization silos. Organizations need to head towards a culture that encourages shared responsibility with a common goal of improving customer experience.

It is no secret that most IT organizations are seldom designed for speed, agility, or innovation. Large companies are hindered by legacy systems, practices, tools, high-handed governance and compliance regimes, archaic, siloed organizational structures, and repressive cultural inertia that impedes change and evolution.

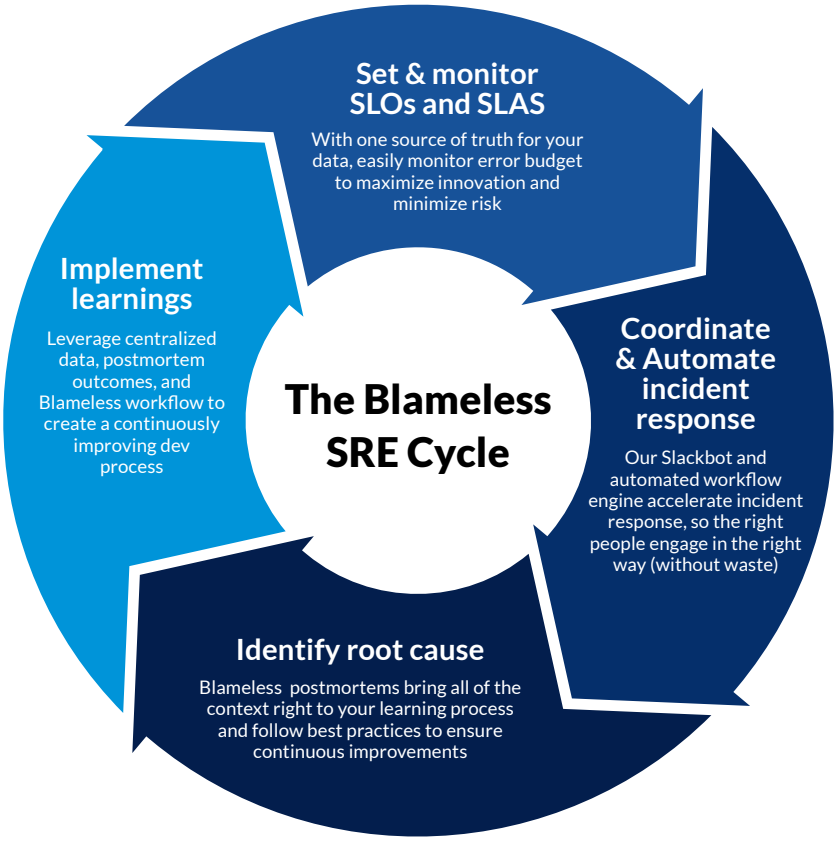
This is one of the major impediments to achieving speed, innovation, and agility at scale. There is a persistent struggle in all large organizations between the desire to innovate and reliability. The development teams are focused on creating innovation through rapid change and experimentation. On the other hand, the operations teams are focused on creating reliability and stability. By adopting a series of cultural changes, this siloed, antagonistic way of working can be remediated.

It's imperative to balance speed to market with system reliability. New features give an organization a competitive edge, but the rate of changes to applications can hamper reliability. This is why the industrialization of SRE must be a multistage process. It's necessary to keep in mind that unstable applications degrade the customer experience while an unhappy customer risks your reputation and profits.



Organizational Learning, Cultural Inertia and Change Management

A revolution of any kind requires the disintegration of the organization so that inefficiency can be removed and ultimately reconfigured. It is the same with SRE as well. Organizations must treat failure as a learning opportunity. System failures happen, and pointing fingers does not help.



A culture of learning from failure helps one understand why things break, how to fix them, prevent or minimize the same thing from breaking again, and rebuild them in better ways. This is where we can encourage the constant evolution of our systems and processes along with the tools. As mentioned before, organizations can no longer play the blame game or pass the buck. Reframing failure successfully is all about a shared sense of purpose and goals. A collaborative approach with shared goals is imperative here.

You're Either an Organization That Learns or You're Losing to the One That Is

Ever since the beginning of the pandemic, we have realized the importance of having a people strategy in place more than just a business-focused one. After all, the health of the system depends on the health of the team supporting it. The leadership needs to support such a cultural shift.

Automation is just a way to streamline tasks. Process improvement and automation can only go so far as the people in the organization. They will have to overcome immense cultural inertia to move ahead. Almost all organizations have inertia. There is an inherent resistance to change, especially in large organizations where the culture may have had years to develop. It permeates across countless practitioners. These practitioners are people, too, and may collectively resist change. Overcoming this is critical.

As time passes, companies develop behaviors. Groups and teams tend to divide up actions and responsibilities along organizational lines. There will be several checks and balances that get established in the name of governance. These may not be related to true governance at all. It is common to see processes that have no reason to exist. Reports get generated that no one reads. No one is willing to do away with them because if they do, bad things can happen. This leads to approval sprawl with a complex labyrinth of 'yes' and 'nos' that are endless and hard to change.

The best way forward is to identify such bottlenecks in the delivery pipeline. Inefficient processes should be removed or replaced, wait times reduced, and unnecessary approvals are done away with.

The reality is that addressing these bottlenecks requires change beyond just traditional SRE practices. One strategy is holding blameless reviews that create global learning and help in maximizing everyone's creativity to find innovative solutions.

Walk the Fine Line Between Innovation and Reliability

Risky activities such as updates and upgrades need to be tracked based on their impact on Service Level Agreements (SLA). One way to walk the fine line between innovation and reliability is to measure it based on Service Level Objectives (SLO).

In combination with SLOs and SLAs, risk analysis is essential when deciding whether a release can be implemented in production or not. Leveraging data, organizations no longer need to engage in back-and-forth arguments and sign-offs to deploy a feature.

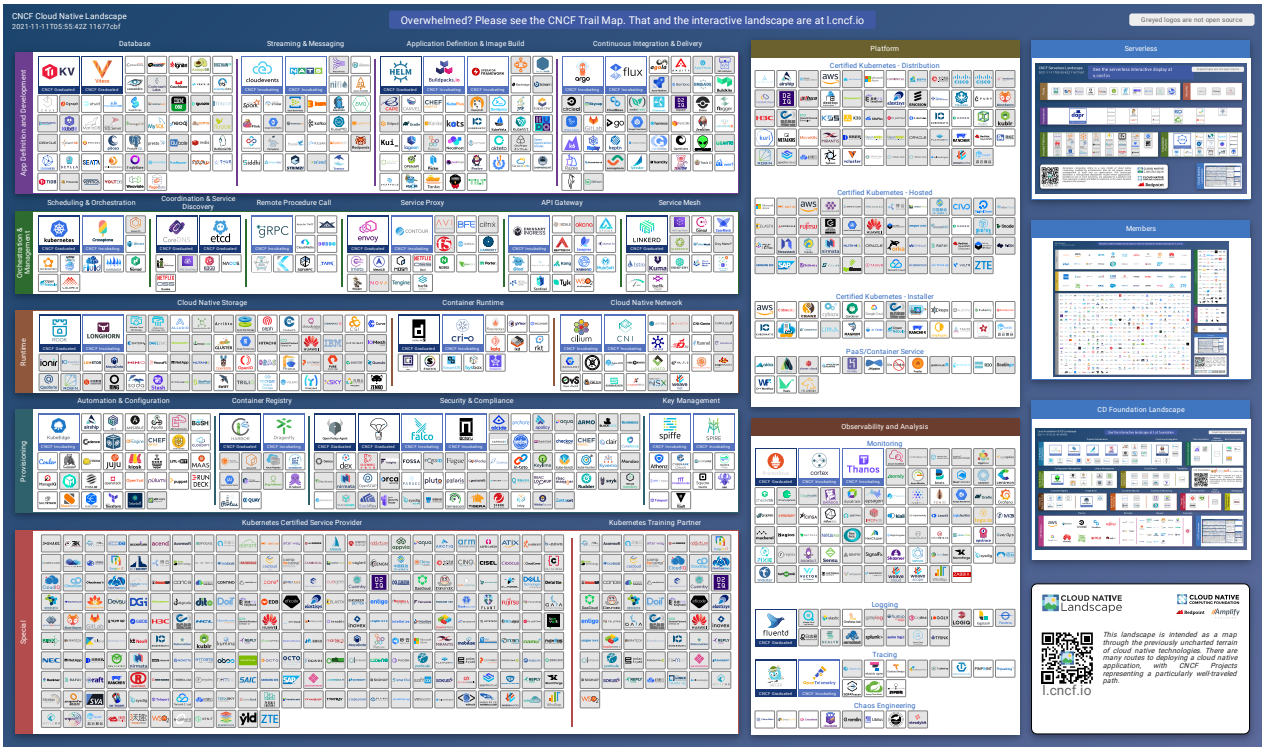
Given that, rapid experimentation must lead to the development of minimum viable products (MVPs) primed for fast delivery and failure to encourage an iterative innovation cycle. At the same time, the preexisting systems need to be industrialized. Hence, they are lean and efficient while delivering business capabilities in an optimized, predictable,
stable, and secure manner, with the ability to scale.

Every process, governance structure, and the system must be continuously and incrementally industrialized to become more stable and reliable. Innovators must seek out new and disruptive business capabilities and avenues to deliver them. At the same time, the industrialized system needs to provide services at a scale that is both lean and efficient.

Empowering Tools With Event and Data-Driven Approach

Organizations need to utilize cutting-edge tools for the continuous evolution of automation. A smart strategy is to automate validation through SLI/SLO-based quality gates. These can help break down monolithic delivery pipelines into event-driven delivery microservice pipelines that take advantage of auto-remediation and self-healing backed by SLO-based safety nets. Here are some of the tools to help with that:

- **Containers and microservices** for creating a scalable system. Examples include Docker (for building and deploying containerized apps) and Kubernetes for container orchestration. Moreover, CI/CD tools like CircleCI, ArgoCD, DroneCI and Jenkins help in implementing the concept of gradual change.
- **Event-driven workflow automation** - automate across multiple tools for complex deployments, self-remediation, and auto-healing. Tools like TriggerMesh, Uber Cadence, Temporal and Stackstorm provides an ITTT (If-this-then-that) methodology to provide a comprehensive workflow automation.
- **Infrastructure as Code** tools to automate everything. Examples include Ansible, terraform, pulumi, etc.
- **Tools for automating functional and non-functional tests in production.** Examples include Selenium, Zephyr, Veracode, etc.
- **Tools for resilience testing** like Gremlin and Chaos Monkey (from Netflix).
- **Monitoring and Observability systems** like Prometheus, ELKStack and DataDog allow metrics-driven continuous monitoring of logs, application metrics/traces, and application performance.



Enterprises also need to leverage SLIs & SLOs in production and as part of continuous delivery improvement. They must implement SLIs/SLOs as a core capability that powers automated quality gates in delivery, performance engineering as a self-service, along auto-remediation as a continuous practice.

The next shift has to be a data-driven decision-making culture. It is key when strategizing about what to invest in — the reliability of services versus new features. Data-driven industrialization of SRE can help depoliticize decision-making while enabling transparency in the decision-making process of delivery in an enterprise.

Measure Everything

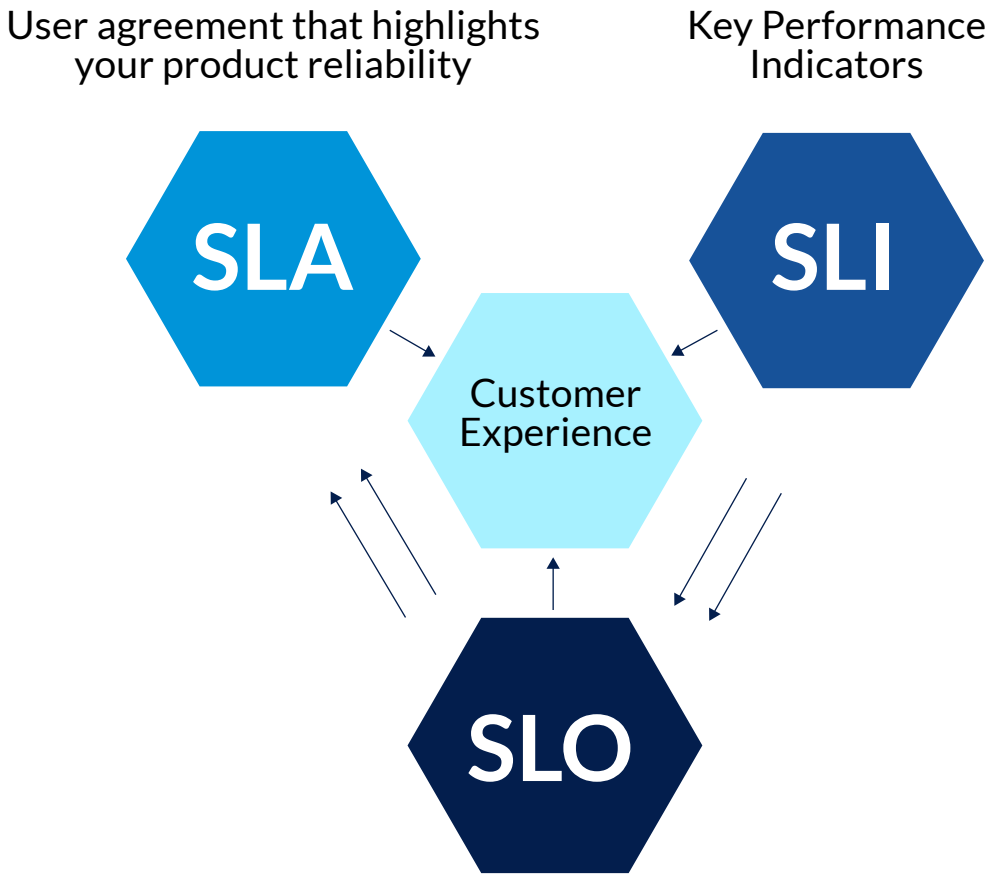
Your system's availability is a non-negotiable precondition to its success. If your services are not available, nothing else matters. Metrics are indeed crucial for any organization looking to walk the fine line between innovation and reliability. Hence, every change must be measured to understand whether the system is bringing the results one expects. It is the only way to minimize reactive IT and move towards the new factory model.

It is highly recommended to leverage the various observability tools available for this purpose and collect Service-Level Indicators (SLIs). SLI is a quantitative measurement of a system's behavior. The prominent SLI for most services is request latency (the time needed to respond to a request), while others are errors per request and throughput of requests per second.

Further, based on the SLIs, stakeholders must set Service-Level Objectives (SLOs). When a system continuously meets these objectives, it is proof that it's reliable. A Service-Level Agreement is then made to promise customers that SLOs will be completed over a specific period.

Having centralized alert automation and collaborative incident response in one solution is necessary if SRE teams want to track deployments and system changes in real-time. This has to work seamlessly across all the engineering functions while consistently monitoring and alerting SLIs across the entire ecosystem of tools in the organization. It is imperative to empower SRE teams with visibility into the developer and IT operations workflows, improve the way SREs manage monitoring and alerting, and help them to build more reliable systems.

We can monitor everything, but the result of doing that is a firehose of information. This can be overwhelming and, more often than not, gets ignored. Instead, organizations need to take the time to thoughtfully consider what metrics to focus on, preventing an overload leading to distraction.



Data-driven Decision Making Phase Comprising of Management, SREs and Developers to Build SMART SLOs

We can monitor everything, but the result of doing that is a firehose of information. This can be overwhelming and, more often than not, gets ignored. Instead, organizations need to take the time to thoughtfully consider what metrics to focus on, preventing an overload leading to distraction.

Let Us Evolve It Paradigms For You

	ITIL	DevOps	SRE
Philosophy & Culture	Align IT with business needs to create a symbiotic relationship Command-and-control and process-driven to mitigate risk	Improve teamwork and eliminate silos Aims to create alignment and minimize silos between development and operations Often oriented toward helping teams improve velocity and quality of deploys	Eliminate toil, design for operability Treats operations as a software problem to maximize efficiency Ideal to support distributed services at scale that need to be hyper-reliable
Key Practices & Tooling	Capacity planning Service catalog / CMDB Problem management Change management / advisory board	Capacity planning On-call Microservices CI/CD Infra as code Monitoring and logging Comms & collaboration	Same as DevOps key practices, as well as Progressive rollouts SLOs & error budgets Observability Chaos engineering

SRE is the distilled version of DevOps implementation principles. Industrializing SRE is the need of the hour for CTOs, product managers, DevOps specialists, and other executives who seek to improve the reliability of their system without sacrificing on innovations.

At iVedha, we automate things, from IT service management to complex SRE workflows. Our development methodologies ensure accurate and timely delivery of services. If you'd like to learn more about iVedha's SRE services, click here to get in touch with us.