DevOps for Enterprise Systems

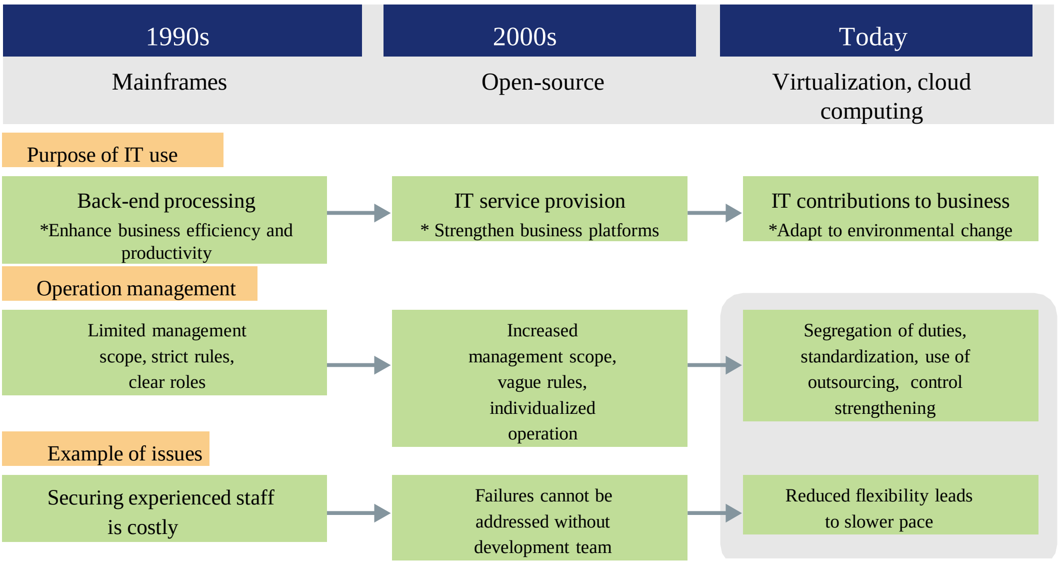
Keys to breaking down barriers between operations and development while maintaining control

This white paper summarizes the purposes of DevOps for enterprise systems and the keys to achieving such purposes.

What is DevOps?

DevOps has garnered attention lately, but it has been given a variety of definitions. Initially DevOps referred to a set of practices, philosophies, and concepts for having developers and operators work together to achieve a faster release of the fruits of development. Today, some people may think of Chef, Jenkins and other open-source automation tools, while others may picture ITIL and other frameworks for process-improvement best practices. This paper will focus on enterprise systems and discuss issues for development and operations and their solutions.

The primary purpose of DevOps, as discussed above, is to use "a unification of development and operations" for rapid delivery of IT services, operation cost savings, enhanced quality, and fewer failures. During the transition from conventional mainframes to open-source systems, control practices for IT overall were strengthened, and clear boundaries were set for the scope of responsibilities for development and operations. With today's virtualization and cloud computing, poor flexibility resulting from the division of work has emerged as an issue. This is because checking the status of systems, approving various requests, and coordinating work for systems changes between development and operations are taking a longer time. This is why DevOps has recently been drawing even further attention.

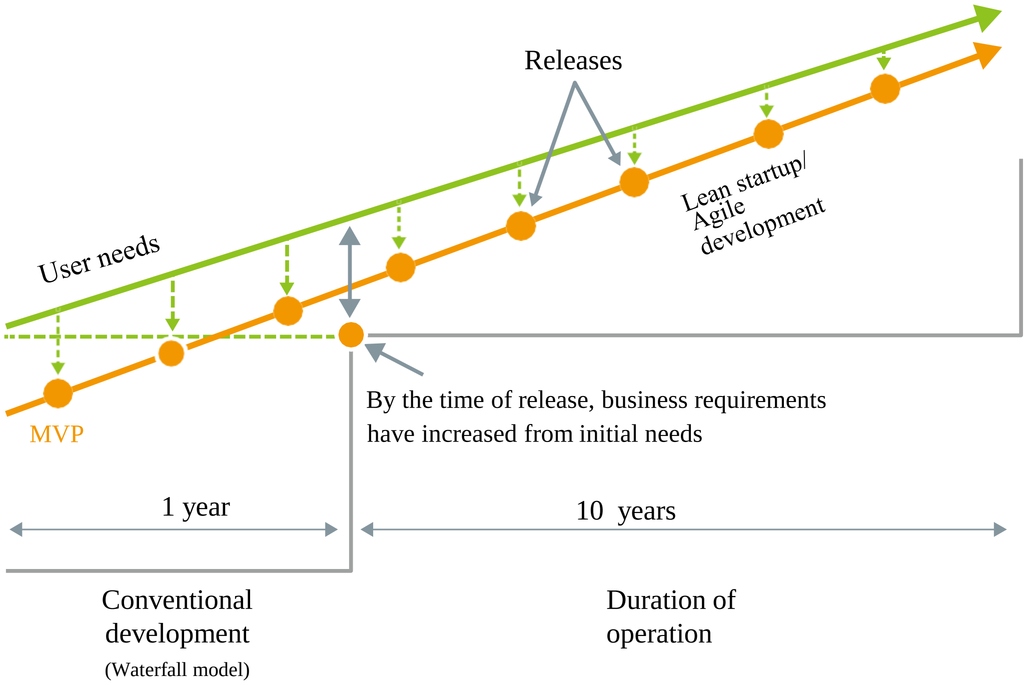


1990s	2000s	Today
Mainframes	Open-source	Virtualization, cloud computing
Purpose of IT use		
Back-end processing *Enhance business efficiency and productivity	IT service provision * Strengthen business platforms	IT contributions to business *Adapt to environmental change
Operation management		
Limited management scope, strict rules, clear roles	Increased management scope, vague rules, individualized operation	Segregation of duties, standardization, use of outsourcing, control strengthening
Example of issues		
Securing experienced staff is costly	Failures cannot be addressed without development team	Reduced flexibility leads to slower pace

Changes in purposes of IT use and operation issues

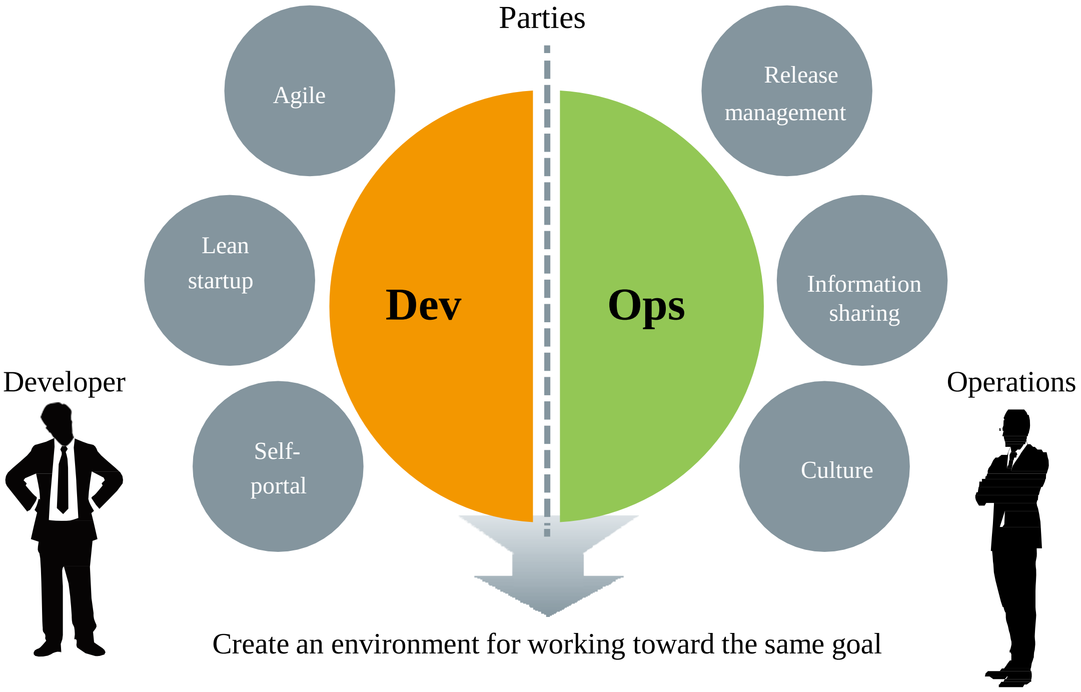
Furthermore, this comes against the backdrop of the shift from the practice of "owning" IT systems, where such systems are developed over a year or two and used for a period of five years or longer, to the practice of "using" IT services, where such services are used on-demand as a minimum viable product (MVP) that is continuously improved.

In terms of development methodology, the conventional waterfall model has given way to agile development. With the waterfall model's long-spanning development process, business requirements tend to increase by the time of release compared to the beginning, resulting in a gap with market needs and wants. For this reason, seeking to achieve speedy development, development teams have started to adopt the lean startup and agile approaches. Still, even if development is finished in a timely manner through agile and other approaches, releases are often delayed; this is because of the work of the dynamics of the operations teams that make them seek to achieve stable operation of the current system with as little change as possible and to maintain a release under a predetermined process.



[Illustration of increased release frequency]

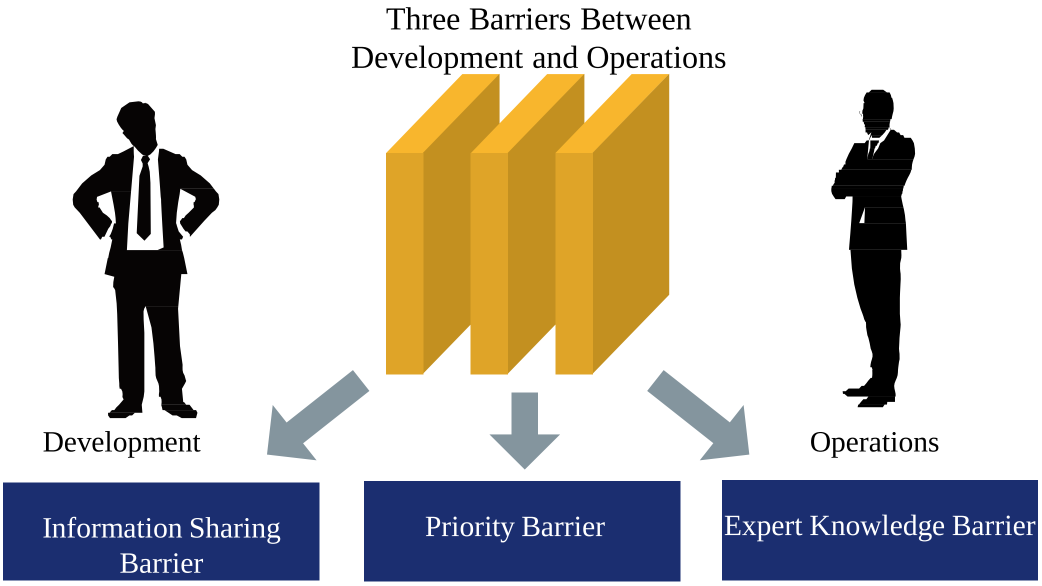
This means the major point of DevOps is creating an environment that brings the development team (which seeks to quickly provide IT services that users want) and the operations team (that wants to maintain the orderliness of IT services, which are increasingly growing chaotic, and achieve secure management) together to work on the same goal.



[The key point is to build an environment for the development and operations teams to work toward the same goal]

Why Unification of Development and Operations is Challenging

What, then, are the specific barriers between the development and operations teams? Setups may differ; in some cases the development and operations teams are in the same department, while in other cases operations are outsourced, but the following three barriers are often cited.



[Three barriers between development and operations]

❶ Information Sharing Barrier: Desired Information Cannot be Obtained Right Away

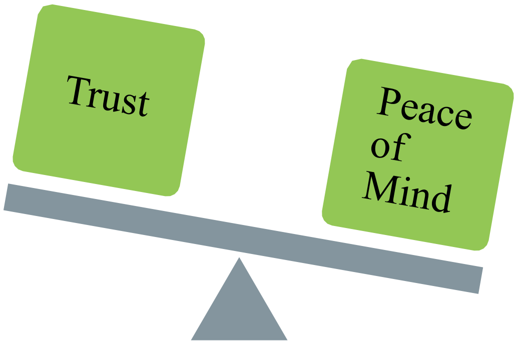
The development team requires a system use history, past log files, and other pieces of accumulated data. Data accumulated over a long period of time is analyzed from different angles, and to this end the development team wants to collect as much data as possible. Conversely, for the operations team, detecting failures and other real-time ascertaining of the situation is important, and thus the team wants to eliminate an overflow of messages to the extent possible and extract the necessary information only. This shows that development and operations use information from the same system differently.

However, not every person on the development team is authorized to access the production environment and therefore, data—including log files—is generally managed by the operations team and supplied upon request from development. For members of the development team, the most efficient approach is to directly access the system for analysis, yet they must ask operations for necessary data as needed. Meanwhile, for members of the operations team, this generates extra work on top of their ordinary workload, increasing their burden. Unless both sides understand the situation and needs, miscommunication may cause things to go wrong, exacerbating the frustration of the two sides.

❷ Priority Barrier: Program Updates and Releases Cannot be Carried Out in a Timely Manner

As discussed above, programs are expected to be released more frequently. However, as flexibly changing virtualization platforms and cloud computing are used more actively, it will become more difficult to ascertain the system as a whole. The operations team does not have a grasp of the details of individual work processes; so even though operations members are responsible for the system platform, they respond to failures of individual systems only as instructed, without addressing the matter flexibly on a case-by-case basis.

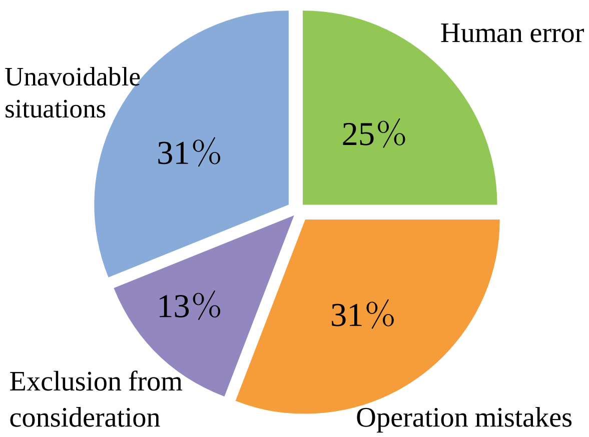
A Nomura Research Institute ("NRI") study of root causes of system failure found that roughly half of breakdowns stemmed from system releases. This finding suggests that system releases should be implemented carefully in a proper process. However, development wants to meet the deadline to win the "trust" of users, while operations wants to guarantee "peace of mind" for the system as a whole and does not want to give special treatment to any single case, resulting in a wide gap between the two.



User trust or peace of mind for the entire system?

③ Expert Knowledge Barrier: Special Treatment Increases Costs and Risk of Failure

The need for standardization of operations has been widely recognized. Yet only a small number of systems are able to standardize everything perfectly, and often special handling is "covered by Operations." Special treatment may require knowledge beyond one's expertise, which likely means higher costs and increased risk of failure.

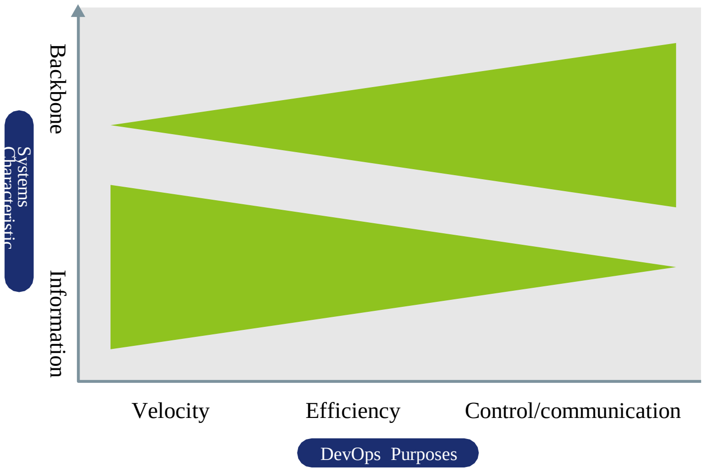


Combined 44% of failure is caused by the "covered by Operations" stance

An NRI analysis of causes of system failure found that human error accounted for just one-quarter of the total, while operation mistakes, such as instruction and application mistakes, and exclusion from consideration due to insufficient examination at the time of platform design made up a combined 44%. These numbers can be reduced with operation standardization from the time of system design; but in reality, little progress has been made in such standardization.

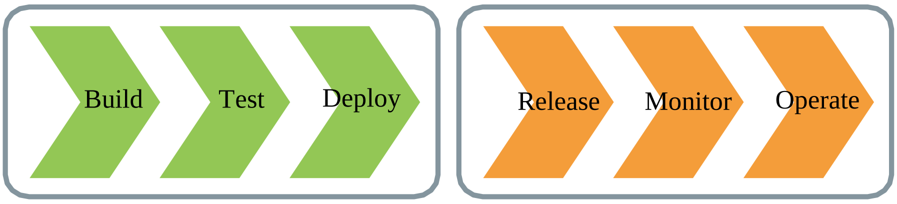
Purposes of DevOps for Enterprise Systems

As discussed above, there are several barriers between development and operations, and demand is rising for DevOps as a solution. However, the main points differ depending on the scale and characteristics of systems. Information systems require velocity due to frequent system updates, while backbone systems place more weight on communication between development and operations, as well as control.



The purposes of DevOps differ depending on different characteristics of systems

Accordingly, the main points of work in DevOps differ between information systems and backbone systems. In order to move the build-test-deploy process at high velocity and high frequency for information systems, the automation of these steps will be important. This is because small modules are repeatedly developed and thus automation is more efficient than manual work. On the other hand, to strengthen cooperation between development and operations in backbone systems, it will be important to enhance efficiency of communication that ensures control and allows for smooth partnership. That is to say, while many steps are taken before information systems are released, with backbone systems, importance is placed on the collaboration between development and operations after the release.



Information systems DevOps (automation) Backbone systems DevOps (information sharing)

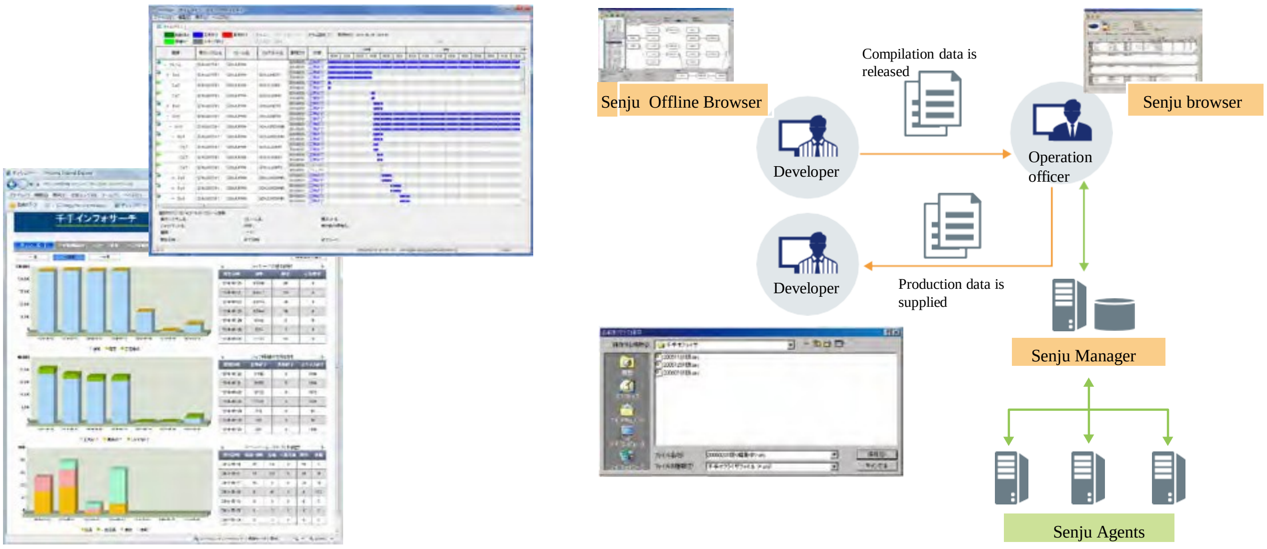
Different system characteristics require different points for achieving DevOps

Senju Family DevOps for Enterprise Systems

Lastly, we will discuss the DevOps solutions that our operation management tool Senju Family provides for enterprise systems with respect to the three barriers above.

1 System Information Sharing Platform and Offline Job Definition Settings

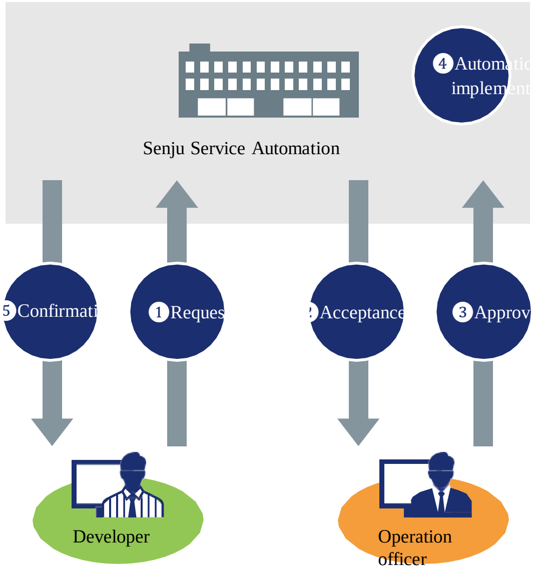
Senju Operation Conductor allows users to securely share event information in the production environment on a web base, job operation history, composition information collection history, setting files, and other information. This enables members of the development team to find and immediately obtain information they want. In an offline environment, settings may be made for job definitions, runbook automation and the like. Instead of the development team performing job design and the operations team having the job design be reflected in an operations management tool, data from development’s job design can be automatically reflected in the tool. This will eliminate errors and misunderstandings that may be caused during manual registration, for instance.



Job Operation History/ Operation Statistics Dashboard / Offliser

❷ Systematization of Request → Approval → Implementation

To follow a proper process in a system release, information sharing with concerned parties and the approval process are necessary. Email-based communication is susceptible to letting something slip through the net or instruction mistakes. Senju Service Automation seamlessly systematizes the series of process from receiving work order to approval, implementation instructions, automated implementation, progress check, and completion check. This enables program updates and releases in a timely fashion.



Senju Service Automation

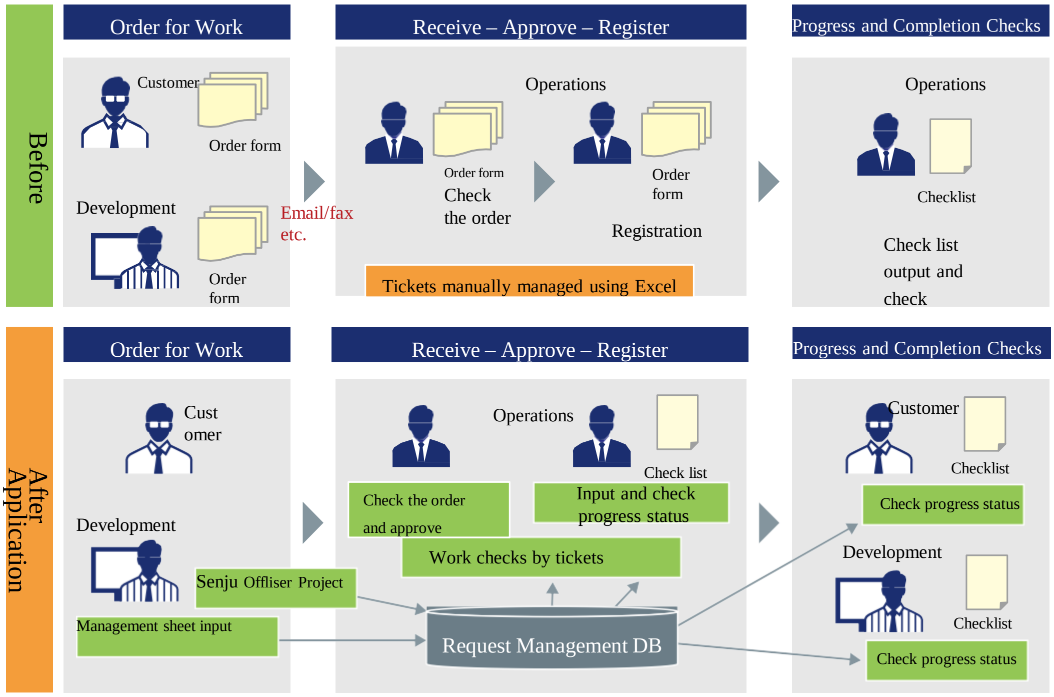
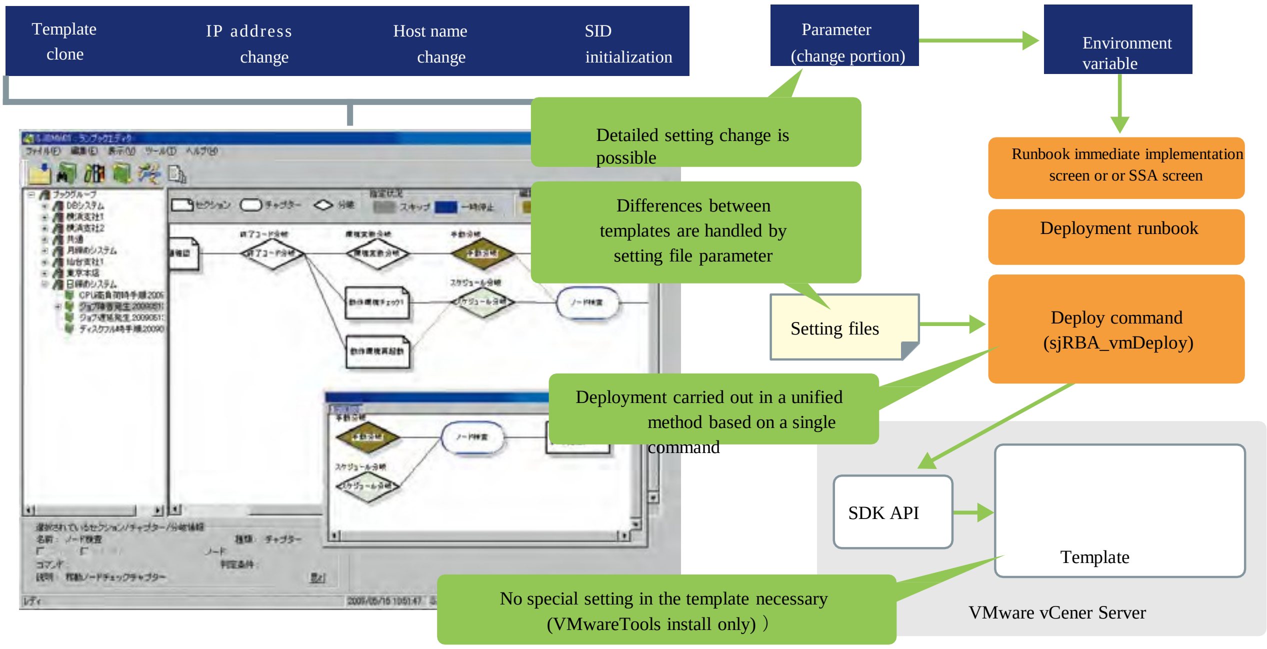


Illustration of Systematization from accepting request to approval, automated implementation and confirmation

❸ Runbook Automation for Manual Systematization and Work Automation

One of the reasons for having special treatment "handled by operations" is an insufficient manual. Any work step not set forth in a manual will have no standardization, lowering the quality of operations. Runbook automation systematizes not only routine work but also special work as work in a manual. Determinations that previously required human judgments can be made autonomously depending on each situation to perform processing. For instance, as the flowchart below shows, virtual machine settings may be changed from the VMware template for each parameter. Such automated work eliminates operation errors and mistakes of exclusion from consideration, and helps achieve speedier handling.


Template clone
IP address change
Host name change
SID initialization
Parameter (change portion)
Environment variable
Detailed setting change is possible
Differences between templates are handled by setting file parameter
Runbook immediate implementation screen or or SSA screen
Deployment runbook
Setting files
Deploy command (sjRBA_vmDeploy)
Deployment carried out in a unified method based on a single command
SDK API
Template
No special setting in the template necessary (VMwareTools install only) ）
VMware vCener Server

Example of building a virtual environment using runbook automation

Conclusion

The main purposes of DevOps are to achieve, through “a unification of development and operations,” quicker releases of IT services, operation cost savings, improved quality, and elimination of failures. Previously, the environment surrounding the development side and the operations side was divided by the three barriers to information sharing, priority and expertise. The key to having successful DevOps is to break down these barriers and create an environment where development and operations work together toward the same goal.

When introducing DevOps for enterprise systems, it should be noted that approaches to DevOps are different for information systems, which require update velocity, and backbone systems, which place importance on communication between development and operations as well as orderliness.

Questions? Contact:

Nomura Research Institute, Ltd.
Senju Information Center
E-mail：senjuinfo@nri-itsa.com URL:https://senjufamily.nri.com/

*The features, methodologies for use etc. in this document are examples and do not guarantee quality or actions.
*Company and product names in this document are generally trademarks or registered trademarks of affiliates. Symbols such as ©, ® and TM are omitted.