



## Enabling DevOps in a Customer-driven Marketplace –The Templated Approach: Part 1

### Practice Head



**Srinivasan Ramasamy**  
VP- Infrastructure &  
Application Services

### Author



**Dhanwandhi Panneerselvam**  
Research Analyst

**aspire**  
SYSTEMS   
*attention. always.*



# CONTENTS

- 1. DevOps in a nutshell**
- 2. DevOps templated approach**
- 3. The 4 dimensions of DevOps**
- 4. Release Management**
  - 4.1 Release management in the era of CI/CD
  - 4.2 A Walk-through of the RM process
- 5. Infra Automation**
  - 5.1 Infrastructure as a code
  - 5.2 A Walk-through of Infra automation
  - 5.3 Use Case
  - 5.4 Benefits of infrastructure automation

# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

## Executive Summary

This is the era of technology where everything moves at lightning pace, DevOps has become the essential part of all enterprises to be more agile enough to respond instantly to the changing market demands. It intends to provide high business value with exceptional service delivery. While enterprises want to design innovative applications they might as well want to resolve internal issues and they assist their customers. This is done by concentrating on the software development and delivery routine without any execution barrier between the development and the operations team. Here, we also talk about maintaining consistency in the cycle through our derived templated approach.

This document lays the foundation and understanding of DevOps culture. It will help you to understand on how you can go about its implementation in your business to achieve maximum benefit.

It focuses on the following key take aways:

- DevOps culture and its effect on your business
- The 4 dimensions of DevOps-
  - Release Management,
  - Infrastructure Automation,
  - Continuous Testing,
  - Orchestration and Feedback loops.
- The well architected templated approach of DevOps dimensions
- How this template is applied across the dimensions
- Real time cases supporting the implementation of the templated approach

The templates of the dimensions were derived through our successful encounters with clients from different industries. This template is worked upon with tools, unique process and control mechanisms across the parameters that influence software development cycle. Thus, a complete walk through of DevOps dimensions will give you clarity on how DevOps implementation can be made simple and efficient to enhance your business development.

This document is geared to **decision makers, DevOps architects, technical team practicing DevOps, who want to explore the DevOps** culture and who are in the process of implementing DevOps.

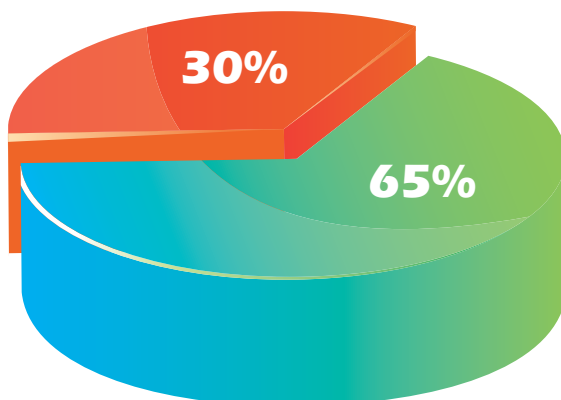
# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

With a huge amount of time, money and energy invested into the process of developing and delivering a product, companies should be meticulous on creating an effective strategy to enhance ROI and customer satisfaction. Any discrepancy with the product due to incompatibility or non-regulation might result in a negative outlook for the product and the company. You can avoid such circumstances by adopting DevOps.

Meeting the needs of the customer in a satisfaction-driven marketplace might not be a cake walk but DevOps implementation can help you do that with ease. Focusing on enabling and aligning business and IT together to speed up delivery while maintaining quality helps to achieve competitiveness and business outcomes. To stay ahead of the contemporaries, it is inevitable to accelerate the interrupted market service and roll out innovations frequently.

**DevOps in a nutshell** is a cross disciplinary process to facilitate every team in the development pipeline and improve the agility of the teams. It helps to establish control mechanism, improve testing, ensure availability and development.

The primary goal of our DevOps approach is to increase the speed of release without altering the quality, automate steps to reduce error and quicken process, establish and control environment readiness while maintaining cost and performance.

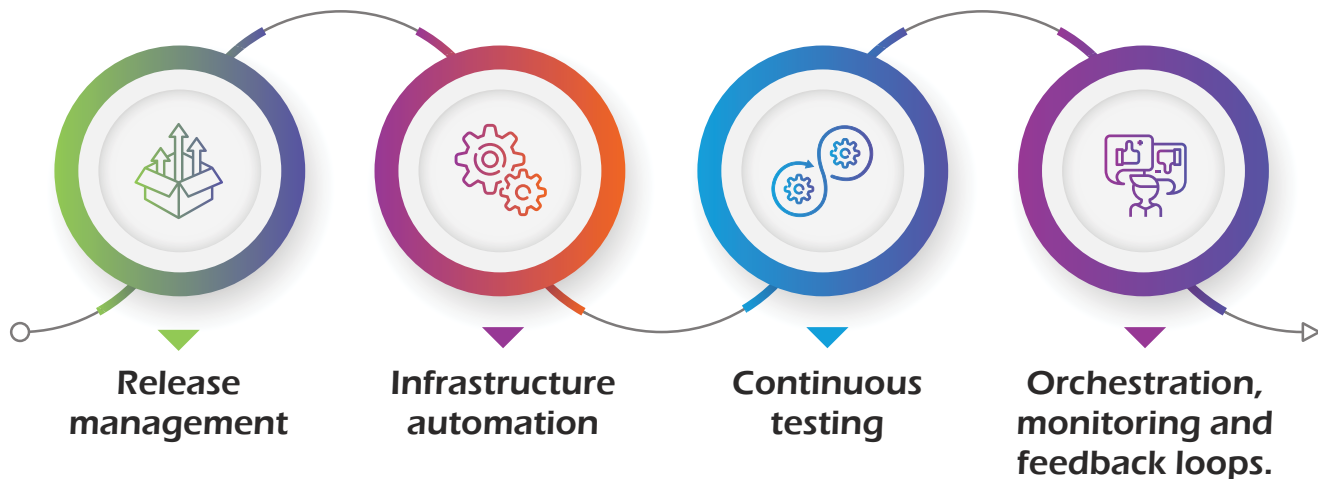


According to the GitLab Global Developer Survey, **65%** of the respondents believe that DevOps workflow saves time during the development process and around **30%** have invested in the previous year, which indicates the steady rise in the adoption of DevOps by enterprises.

# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

## The 4 Dimensions of DevOps

There are four dimensions that very well fit into every scenario, they are



## DevOps Templated approach:

It is a set of parameters that we group into logical categories that are closer to real life cycle development called dimensions. The parameters are significant factor of the cycle affecting the implementation details.

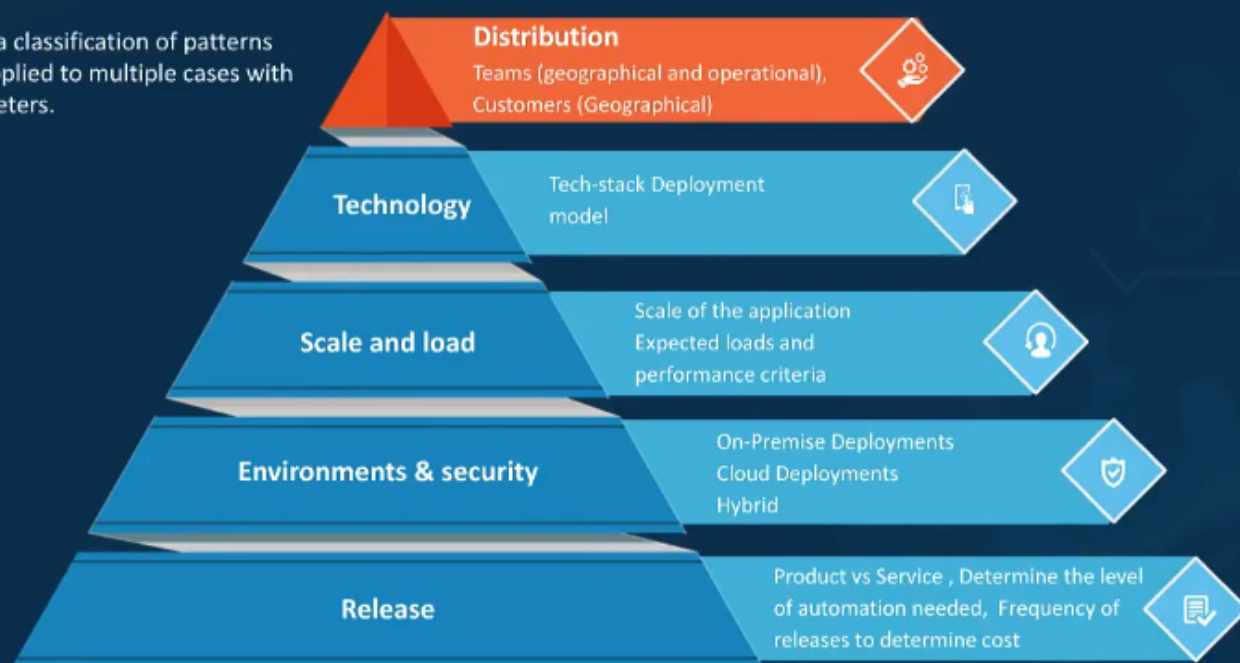
The parameters that influence the dimensions are

- Distribution of your organization geographically and operationally
- Technology models involved in the deployment process
- Present and future scale and load of application with performance
- Environment and security
- Release frequency






Applying these parameters across all dimensions help identify the necessary condition that helps to strategize all functions from development to operations.

# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

A template is a classification of patterns that can be applied to multiple cases with similar parameters.



Our experts have derived the DevOps template as a result of our successful engagements with clients from various sectors. Thus, we use this template to simply break things into small problems and resolve one at a time to avoid chaos, arrive at tools, process and control mechanism for specific requirements.

	Infrastructure Automation	Release Management	Continuous Testing	Orchestration
 <b>Distribution</b>	TOOLS	TOOLS	TOOLS	TOOLS
 <b>Technology</b>				
 <b>Scale and Load</b>	PROCESS	PROCESS	PROCESS	PROCESS
 <b>Environments &amp; Security</b>				
 <b>Release</b>	CONTROL MECHANISMS	CONTROL MECHANISMS	CONTROL MECHANISMS	CONTROL MECHANISMS



# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

## Release Management (RM):

While DevOps encompasses software from the stage of development to delivery into production environment, release management aims at delivering software into its respective environment.

On a wider perspective, release management manages and controls the process as to which software deploys into which environment and at what time with the help of tools like XL Release, Clarive, Tasktop integration hub, octopus deployment, etc. unlike the old days with CCB, CAB.

## Release management in the era of CI/CD

The continuous integration and continuous delivery has made the development process more nimble and eased the way release management works. The traditional way of unilateral release cycle consists of research, planning, development, testing and releases without any feedback loop hence, to make the process more efficient and the releases more rapid CI/CD was introduced.

With continuous integration, you can integrate codes continuously into a central repository through automation. The key aspect of CI is to detect bugs quickly, save validation time, enhance the product quality, improve developers' productivity and give new updates. It eradicates the lone long working hours and introduces changes to the pipeline when the work is done.

The CI expands and extends to continuous delivery concerning the readiness to release the product into the production environment post the build-stage in incremental chunks. It helps in streamlining their releases as the release management ensures the alterations made are ready for release at any point of time. It is more agile and helps in quicker response to information.

The next stage that follows continuous delivery is the continuous deployment method where every changed or patched code goes through the pipeline automatically to fall into production. It aids in quicker return on investments and feedbacks.



# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

The key effects of CI/CD on release management are as follows:

1. The Advent of CI/CD enables the RM to meet the requirements of both development and operations team.
2. With the emergence of CD, the releases are quicker which require the release managers to deal with the front ends to ensure changes required from the customers.
3. Release managers are able to check the release quality from the development stage to production stage.
4. The decision making has become easier with CI/CD intrusion due to automated tasks throughout the pipeline.

With the addition of CI/CD into the release management, the release managers are at higher demand than ever. An in-depth knowledge of CI/CD tools and the workflow is mandatory with the essential inter and intra personal skills to establish the modern work culture.

## A Walk-through of the RM process with the templated approach

We formulated templated approach with the master list of

- **Tools:** Based on our experience on tools like the enterprise release automation tools, release automation for containers, open source tools, Paas providers, we build a set of rules and regulations to perform the templated approach that has the right impact of the ROI.
- **Processes:** It comprises CI,CD,CT, IaC, Paas based delivery, monitoring, incident management and fallback process and these are defined for the development operation process.
- **Control mechanism:** This deals with having control over the number of pipelines associated in a process, how it should merge with components or how components should integrate with one another, approvals, escalations and success conditions

With all these above set of tools, process and mechanisms, we develop a template incorporating all our learnings and experience into rules. We apply the elimination approach to these rules to narrow down the list based on the requirement with specific conditions with respect to technology and cost limitations for validation.

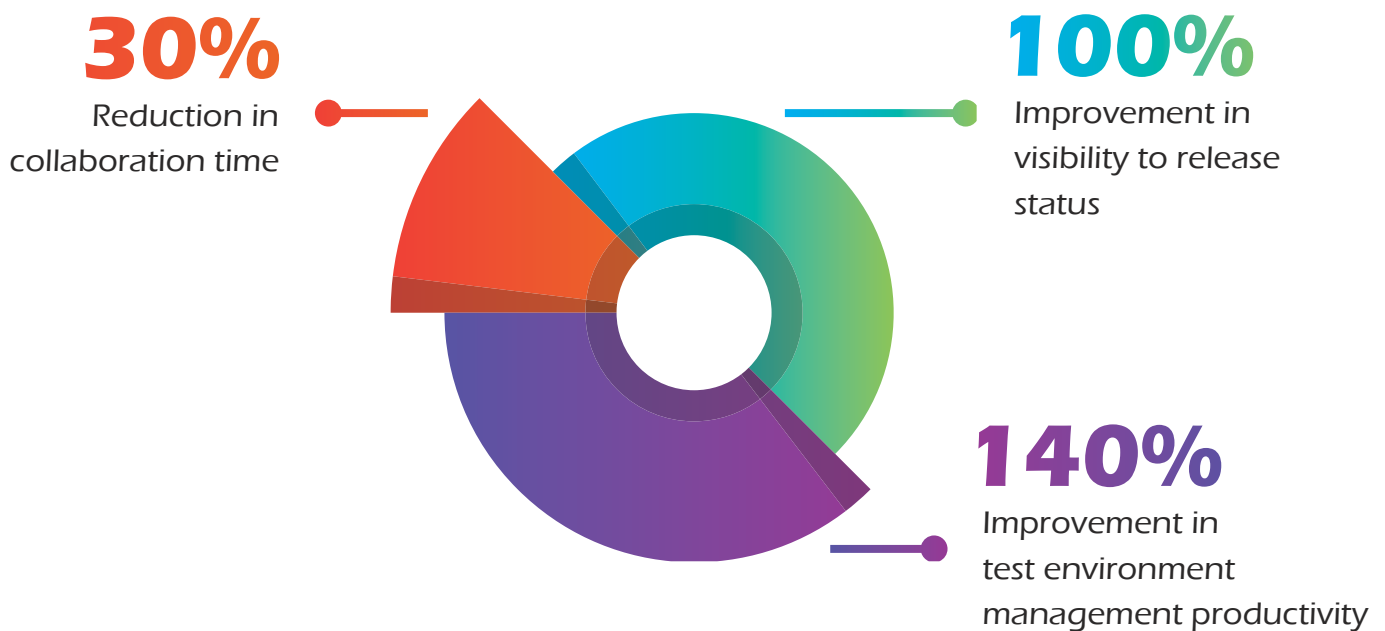


# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

## Use Case

A large pharmacy led retailer in the UK was struggling to maintain the application delivery and seamless user experience with their app. The manual practice with spreadsheets increased the complexities and was divergent in meeting the needs of various portfolios also; there was no visibility of test environment requirement over time.

Incorporation of release management led to



## RM's advantages on business

The essential Release management in a DevOps culture on practice fetches the following

- Paves way for an uninterrupted release flow
- Automated release management helps in Consistent release with quality output
- Provides a niche for change adaptation
- Maintains Strong auditing
- Zero downtime
- Saves time and cost

# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

## Infra Automation

Infrastructure automation is the process of setting up the environment for smooth operations like installing OS, configuring servers on instances and their communication with the software for enhanced usage and faster deployment with lesser error. It eliminates the human error in IT configuration and tracking and saves the task time of administrators.

Let's look at the components of the infrastructure management and automation

- Configuration management maintains configuration, code and scripts that are required to provision the application in any environment
- Configuration provisioning deals with deploying, scaling and monitoring the hardware that includes roll back capabilities
- Infrastructure orchestration manages infrastructure automatically, i.e. The configuration management and provisioning come together to manage an application with the help of orchestration
- Log management, monitoring and custom scripts ensure to achieve adhoc based automation like up-start scripts, init, etc.

The infra automation process uses various tools like Chef, Jenkins, Puppet, Ansible, Docker that derives the success of its implementation and touts as the solution for all deployment inconsistencies.



# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

## Infrastructure as a code :

Infrastructure as code (IaC) is the conception of managing the operation environment like how you manage your applications and codes for releases. In order to make infrastructure changes, you need not make any manual changes to the configuration or the one-off scripts instead; you need to manage the operations' infrastructure. To be precise, you treat infrastructure like code.

You can write machine readable files and deploy them to support the releases and other IT processes. You can also describe IaC as a programmable infrastructure and it is critical for DevOps culture which relies on automated workflow.

There is a need for a replica of an environment to test the codes that are same as that of codes deployed to the live environment. Such instances ensure non collision of new and existing codes to avoid error generation.

In the earlier days, the system administrators set up a physical server to mirror the live machine supporting the production environment followed by database support check by database administrators and they sign it off to the testing team that runs several tests. Thus, a code passes through all these levels before deploying into the production. However, you can't assure an error free output despite the long process as manual recreation of environments is prone to potential error.

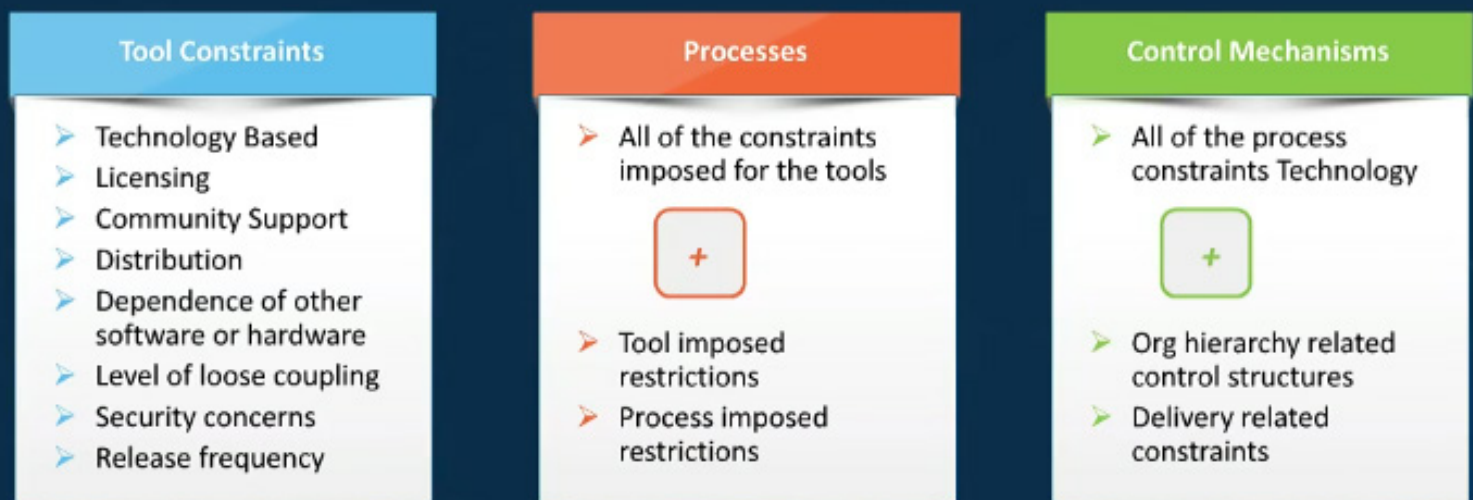
Today, the developer writes codes and configuration instructions that initiate action from environments. The configuration management creates a virtual test environment with server and database that replicates the live environment in every aspect; also it transfers the live data to the virtual test platform. Upon the creation and transition of environment and data, a set of tools execute tests to check compliance, detect error and resolve the issue. Further,

the code is set for deployment to the live environment. IaC ensures that if the code in the replicated environment works well, it will exactly behave the same in the live environment. A compendious variety of tools are available to practice the provisioning and configuring process like Chef, Puppet, Ansible, Terraform, AWS cloud formation, Google cloud deployment manager, etc. The choice of tools used differs for every individual infrastructure as it depends on the specific functionalities.

# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

## A walk through of Infra automation through templated approach

We maintain a knowledge base of tools around which we create rules subjective to the requirements and it enables the elimination process with all parameters.



A simple elimination process with each parameters gives you a finite list of tools, process and control mechanism. If the elimination process does not provide a single tool, you can use more than one tool to enable the required features with respect to the cost preference.

## Use Case

Let consider a product development scenario wherein two sets of development takes place with the initial development stage and the second phase of customer integrated development cycle.

Here is a list of tools available in the market for all the categories of infra automation that we use for the application on the templated approach:



# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

Configuration Management	Configuration Provisioning	Infrastructure Orchestration/Templates	Monitoring/Log Management
<ol style="list-style-type: none"> <li>1. Chef</li> <li>2. Puppet</li> <li>3. Ansible</li> <li>4. Salt</li> <li>5. Rudder</li> <li>6. Juju</li> <li>7. MicroSoft SCCM</li> <li>8. CFEngine</li> <li>9. Opswork</li> <li>10. Azure Automation</li> <li>11. Turbonomic</li> <li>12. Foreman</li> <li>13. Packer</li> </ol>	<ol style="list-style-type: none"> <li>1. Terraform</li> <li>2. Cloudformation</li> <li>3. ARM</li> <li>4. Foreman</li> <li>5. Vagrant</li> <li>6. Powershell DSC</li> <li>7. Turbonomic</li> <li>8. TFS/VSTS (Vsphere SDK)</li> <li>9. Octopus</li> <li>10. Kubernetes</li> <li>11. Docker Swarm</li> <li>12. Rancher</li> <li>13. ECS (Elastic Container Service)</li> <li>14. AKS (Azure Container Service)</li> <li>15. Beanstalk</li> <li>16. Webapp</li> <li>17. Cloudfoundry</li> <li>18. Google Engine</li> </ol>	<ol style="list-style-type: none"> <li>1. AWS ServiceCatalog</li> <li>2. Azure Service Catalog</li> <li>3. ManagelQ</li> <li>4. Cloudify</li> <li>5. Foreman</li> <li>6. Kubernetes</li> <li>7. Docker Swarm</li> <li>8. Rancher</li> <li>9. Openshift</li> <li>10. Jenkins</li> <li>11. ServiceNow</li> </ol>	<ol style="list-style-type: none"> <li>1. Nagios</li> <li>2. Zabbix</li> <li>3. Monit</li> <li>4. Cacti</li> <li>5. OpManager</li> <li>6. Newrelic</li> <li>7. Dynatrace</li> <li>8. LogicMonitor</li> <li>9. CloudWatch</li> <li>10. Azure Monitor</li> <li>11. App insight</li> <li>12. Scalyr</li> <li>13. Pagerduty</li> <li>14. Webmatrix</li> </ol> <ol style="list-style-type: none"> <li>1. Splunk</li> <li>2. logstash</li> <li>3. LogRhythm</li> <li>4. Graylog</li> <li>5. Sumologic</li> <li>6. Loggly</li> <li>7. papertrail</li> </ol>

Applying the parameters across the categories, we filter the tools used in each category that best suit the business needs.

<b>➤ Distribution</b> <ul style="list-style-type: none"> <li>❑ 7 different teams + END users in multiple geographies</li> <li>❑ Multiple Geographic location</li> </ul>	Configuration Management	Configuration Provisioning	Infrastructure Orchestration/Templates	Monitoring/Log Management
	<ol style="list-style-type: none"> <li>1. Chef</li> <li>2. Puppet</li> <li>3. Ansible</li> <li>4. Salt</li> <li>5. MicroSoft SCCM</li> <li>6. CFEngine</li> <li>7. Opswork</li> <li>8. Azure Automation</li> <li>9. Turbonomic</li> <li>10. Foreman</li> <li>11. Packer</li> </ol>	<ol style="list-style-type: none"> <li>1. Terraform</li> <li>2. Cloudformation</li> <li>3. ARM</li> <li>4. Foreman</li> <li>5. Vagrant</li> <li>6. Powershell DSC</li> <li>7. Turbonomic</li> <li>8. TFS/VSTS (Vsphere SDK)</li> <li>9. Octopus</li> <li>10. Kubernetes</li> <li>11. Docker Swarm</li> <li>12. Rancher</li> <li>13. ECS (Elastic Container Service)</li> <li>14. AKS (Azure Container Service)</li> <li>15. Beanstalk</li> <li>16. Webapp</li> <li>17. Cloudfoundry</li> <li>18. Google Engine</li> </ol>	<ol style="list-style-type: none"> <li>1. AWS ServiceCatalog</li> <li>2. Azure Service Catalog</li> <li>3. ManagelQ</li> <li>4. Cloudify</li> <li>5. Foreman</li> <li>6. Kubernetes</li> <li>7. Docker Swarm</li> <li>8. Rancher</li> <li>9. Openshift</li> <li>10. Jenkins</li> <li>11. ServiceNow</li> </ol>	<ol style="list-style-type: none"> <li>1. Nagios</li> <li>2. Zabbix</li> <li>3. Monit</li> <li>4. Cacti</li> <li>5. OpManager</li> <li>6. Newrelic</li> <li>7. Dynatrace</li> <li>8. LogicMonitor</li> <li>9. CloudWatch</li> <li>10. Azure Monitor</li> <li>11. App insight</li> <li>12. Scalyr</li> <li>13. Pagerduty</li> <li>14. Webmatrix</li> </ol> <ol style="list-style-type: none"> <li>1. Splunk</li> <li>2. logstash</li> <li>3. LogRhythm</li> <li>4. Graylog</li> <li>5. Sumologic</li> <li>6. Loggly</li> <li>7. papertrail</li> </ol>



# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

## ➤ Distribution

- ❑ 7 different teams + END users in multiple geographies
- ❑ Multiple Geographic location

### Configuration Management

1. Chef
2. Puppet
3. Ansible
4. Salt
5. MicroSoft SCCM
6. CFEngine
7. Opswork
8. Azure Automation
9. Turbonomic
10. Foreman
11. Packer

### Configuration Provisioning

1. Terraform
2. Cloudformation
3. ARM
4. Foreman
5. Vagrant
6. Powershell DSC
7. Turbonomic
8. TFS/VSTS (Vsphere SDK)
9. Octopus
10. Kubernetes
11. Docker Swarm
12. Rancher
13. ECS (Elastic Container Service)
14. AKS (Azure Container Service)
15. Beanstalk
16. Webapp
17. Cloudfoundry
18. Google Engine

### Infrastructure Orchestration/ Templates

1. AWS ServiceCatalog
2. Azure Service Catalog
3. ManagelQ
4. Cloudify
5. Foreman
6. Kubernetes
7. Docker Swarm
8. Rancher
9. Openshift
10. Jenkins
11. ServiceNow

### Monitoring/Log Management

1. Nagios
2. Zabbix
3. Monit
4. Cacti
5. OpManager
6. Newrelic
7. Dynatrace
8. LogicMonitor
9. CloudWatch
10. Azure Monitor
11. App insight
12. Scalyr
13. Pagerduty
14. Webmatrix
1. Splunk
2. logstash
3. LogRythm
4. Graylog
5. Sumologic
6. Loggly
7. papertrail

## ➤ Distribution

- ❑ 7 different teams + END users in multiple geographies
- ❑ Multiple Geographic location

## ➤ Technology

- ❑ Java Stack
- ❑ Application Code Deployment (IAAS)

## ➤ Scale and Load

- ❑ 10000 Concurrent users

### Configuration Management

1. Chef
2. Puppet
3. Ansible
4. Salt
5. CFEngine
6. Opswork

### Configuration Provisioning

1. Terraform
2. Cloudformation
3. ARM

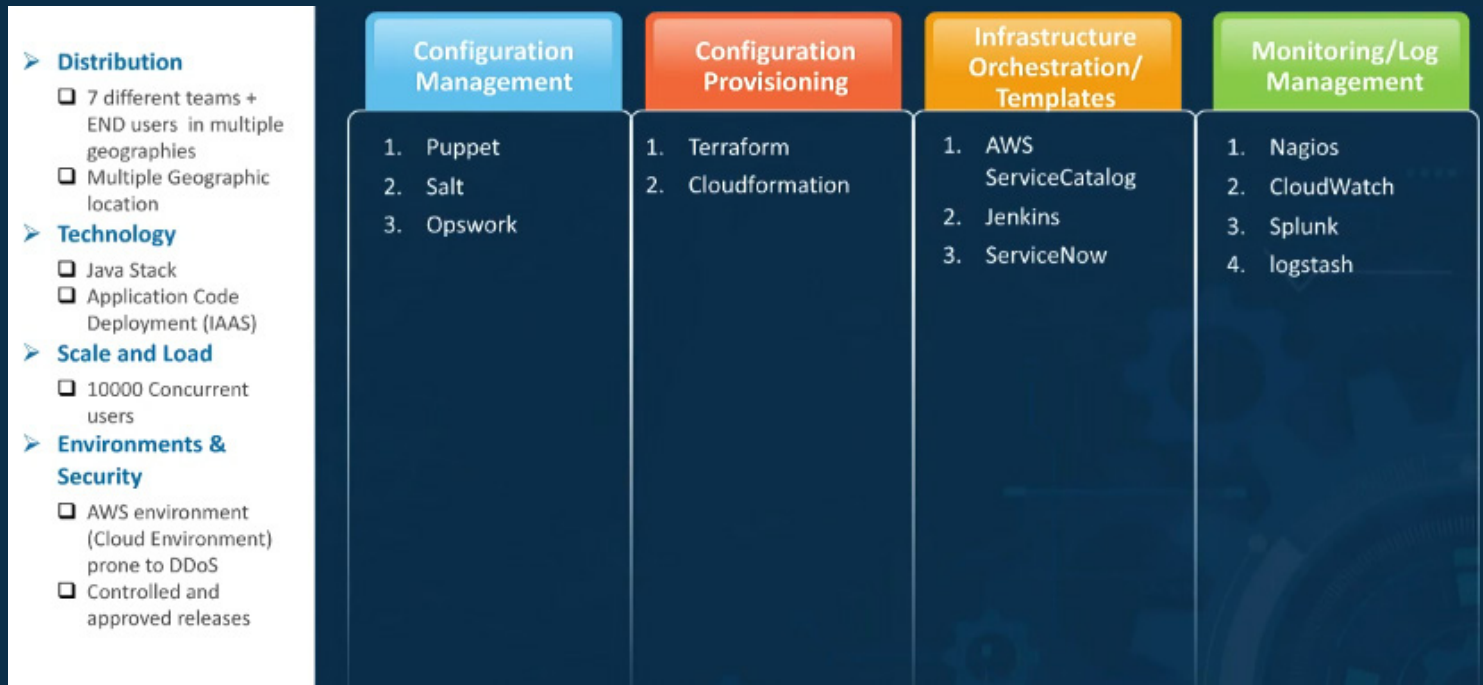
### Infrastructure Orchestration/ Templates

1. AWS ServiceCatalog
2. Azure Service Catalog
3. ManagelQ
4. Cloudify
5. Jenkins
6. ServiceNow

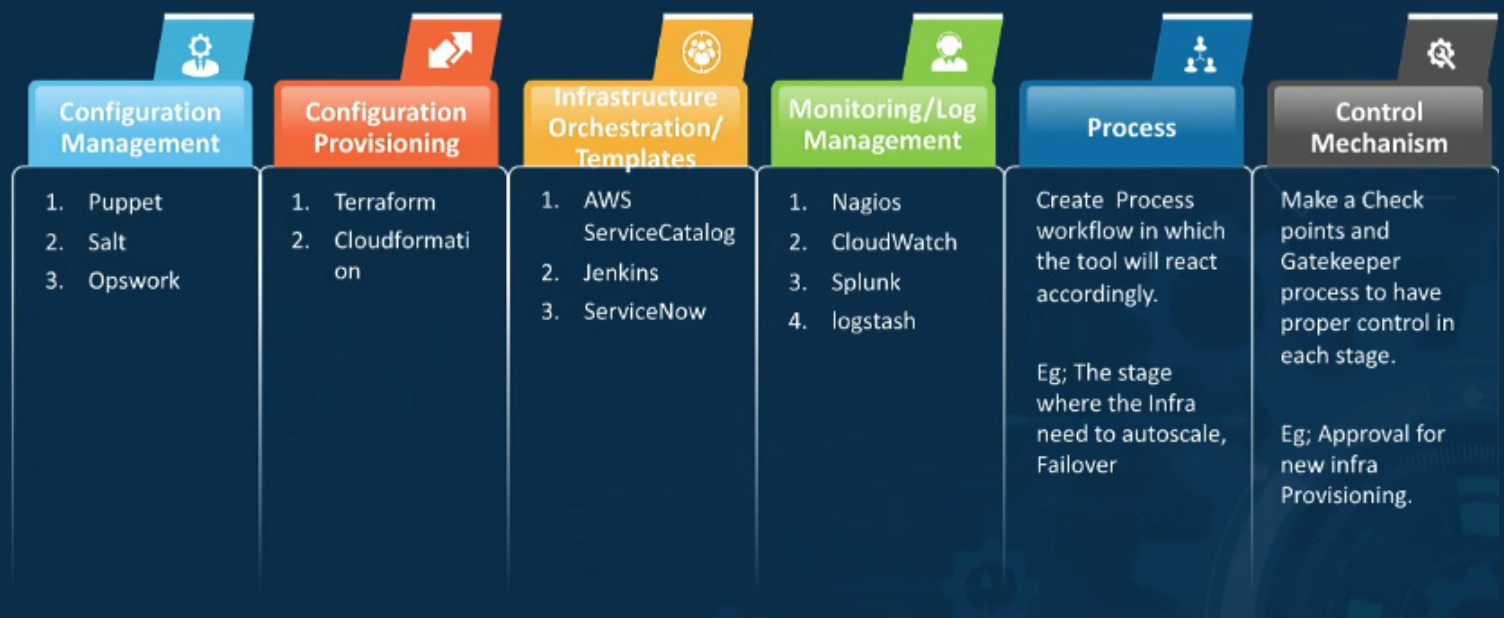
### Monitoring/Log Management

1. Nagios
2. Zabbix
3. Cacti
4. Newrelic
5. Dynatrace
6. CloudWatch
7. Azure Monitor
8. App insight
9. Scalyr
10. Pagerduty
1. Splunk
2. logstash
3. Sumologic

# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach



We define and create a process workflow for the tools to react as and when the infra is autoscaled. The control mechanism acts as a point of approval where a team approves the process. Thus the templated approach provides us with the finite approach to help in all aspects.



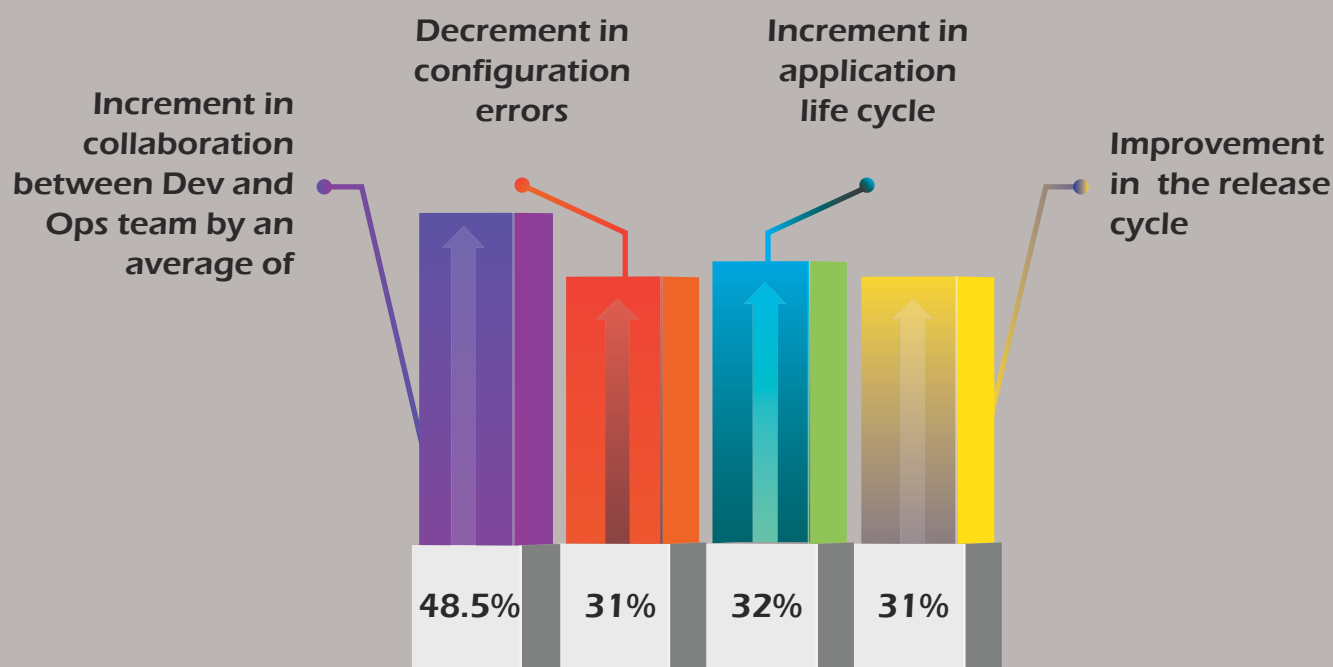
# Enabling DevOps in a Customer-driven Marketplace – A Templated Approach

## Benefits of infrastructure automation

There are several advantages in implementing infrastructure automation in an organization. Let's look at the most significant ones among them.

- Increases efficiency with high security
- Reduces human resource costing without affecting the product quality
- Improves stability, reliability and agility as organization become complaint at all levels of SDLC
- Enhances cross team collaboration on large-scale projects
- Improves resilience and ensures good customer experience

According to the commissioned study conducted with 150 professionals by the Forrester consulting, by utilizing IaC you can achieve



The above two dimensions gives us a wide knowledge of DevOps process and how it helps organizations to improve release cycles with better scalability and reliability. On the verge of enhancing the software development life cycle, we have formulated the templated approach across these dimensions of DevOps. We shall learn in detail about the other dimensions of DevOps in our next article.



## ABOUT ASPIRE

- Global technology services firm with a global presence across North America, Europe, APAC and Middle East
- Specific areas of expertise around Software Engineering, Digital Services, Testing and Infrastructure & Application Support
- Vertical focus among Independent Software Vendors, Retail, BFSI and Education
- 3000+ employees; 160+ active customers
- CMMI Maturity Level 3, ISO 9001:2015 and ISO 27001: 2013 certified
- Presence across Singapore, US, UK, Netherlands, Middle East and India
- Recognized 9 consecutive times as “Best Place to Work for” by GPW Institute