Exastro

The keywords that promotes Cloud nativeness

What is "Offensive and Defensive Automation"?

Just what is hindering Cloud nativeness?

In order to keep up with the changes in the business environment, many companies are asking for propulsion of DX (Digital Transformation). The technology that is the key for doing so is Cloud Nativeness.

Different from the current systems with monolithic architecture, Cloud nativeness makes it possible to make systems both scalable and open.

However, Developing cloud native applications while operating on a monolithic system is far from easy. This leads to many IT engineers hanging on to the current systems, which ultimately creates a demand for capable engineers.

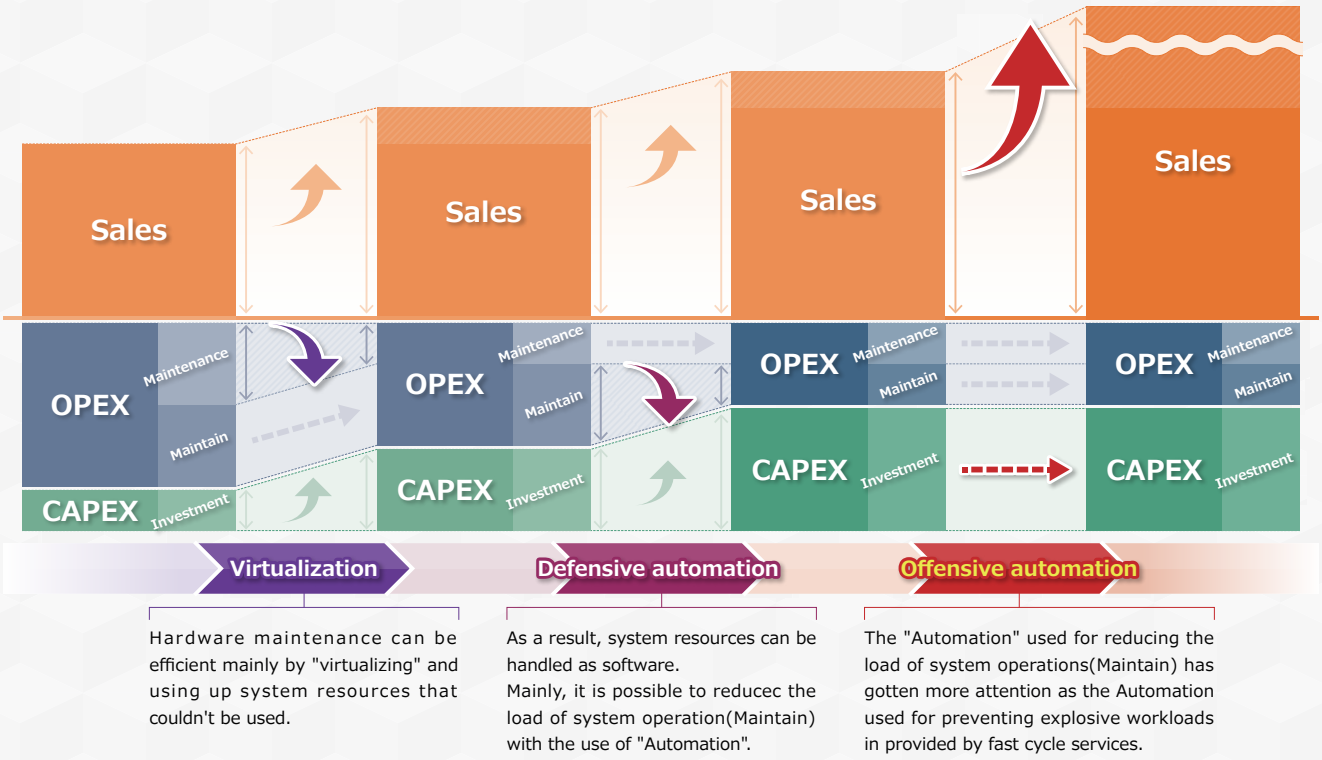
In order to solve this problem, we need to free the IT Engineers who are working on, constructing, operating and still holding onto the current systems. Additionally, it is necessary to secure enough IT Engineers capable of operating and creating cloud native systems.

However, it' s not as easy as only saving labor by automating the construction and operations of already monolithic systems. Cloud native systems are also in need of automated construction and operation in order to provide a speedy service development.

Two different types of automation with two different goals.

In the world of systems, the word" Automation" has been brought up as a keyword for 4-5 years. The first progression of Automation was "Virtualization". With the help of virtualizing, it became possible to cut maintenance costs by dealing with hardware releases in a more systematic and effective manner.

As the next step, the maintenance costs getting cut, was getting attention. They automated the construction and operation in



the current monolithic systems, which we now call Defensive Automation. By replacing many operation and construction tasks in systems and automating them, people were able to cut both labor hours and human resources by making them more effective.

The next step after the progression to Cloud native systems with the use of "Defensive Automation" is "Offensive Automation".

Offensive Automation is the initiative of operating and constructing cloud native systems faster. Automatic construction and operation, as well as reconstruction of environments using Blue-Green Deployment will become standard. The goal of "offensive automation" is raising the Return on Investment (ROI).

Both "Defensive" and "Offensive" automation has often been thought of having different use cases with their automation techniques, resulting them to be discussed in separate areas.

However, the truth is that even if cloud nativeness progresses, there are many systems that will use Monolithic and Cloud native hybrid systems for constructing and operating.

Therefore, it will be important to make sure to choose the carefully implemented techniques of Offensive automation after lowering the costs of the current operation costs by using defensive automation.

What should you keep in mind when implementing "Defensive Automation"?

There are several different types of work and operations that should be automated within "Defensive Automation" such as the automation of Construction and Regular operations.

However, the one that should be prioritized is the automation of countermeasures for obstacles and problems.

Countermeasures for obstacles and problems are operations that are often individualized and requires high capacity. Meaning

that being just one minute late can have a large effect on the service.

It is therefore essential to shorten the labor hours and making human recourses more efficient.

Then what should be taken into consideration when automating countermeasure for obstacles? Let`s take a look at the 5 steps of countermeasures against problems.

1 . Obstacle detection
2 . Temporary countermeasures
3 . Check effect on service
4 . Investigation
5 . Normal Countermeasures

1 . Obstacle detection

The first step when an obstacle appears is to become aware of the problem. As this step consists of having specialized software to circulate the system as monitoring programs, it can be considered the easiest step to automate.

2 . Temporary countermeasures

After recognizing the obstacle, it is important to keep the damages/effects it might have to the minimum the moment it is found. The method of the countermeasure varies depending on the obstacle. What usually happens in real life scenarios is that the expert/leader takes the lead and deals with the problem manually.

As you might imagine, it is often work that is both individualizing and requires speed, making it one of the most important steps to automate.

However, it can include methods such as notifying the users and continuing the service by using alternative methods.

3 . Check effect on service

This is the step for checking the grade of the effect the obstacle has had on the service. It is a step needed in when comparing it to SLA (Service Level Agreement) and when reporting to the stakeholders.

This step is often not considered when thinking of what should be automated, but as it is a fundamental step in the operating process, it is possible to greatly reduce the operating workload by incorporating it into the automated process.

4 . Investigation

In the case of a new unknown problem, there is a need to know why and how it happened. The investigation usually consists of checking and analyzing the system resource status and the system output log.

Most of the focus here is on the manual work. However, it is possible to automate the operation of collecting information.

While not all of it, by automating parts of the operation, we can make the focus less on "Something that can only be done manually" and more on how to be more resourceful".

5 . Fundamental countermeasures

Lastly, countermeasures created to prevent the same problem from appearing. This step should be done automatically or manually depending on how complicated the countermeasure should be.

That being said, it is possible to modularize the countermeasure and automate tasks that have high reuse factors.

There are the steps of obstacle countermeasures.

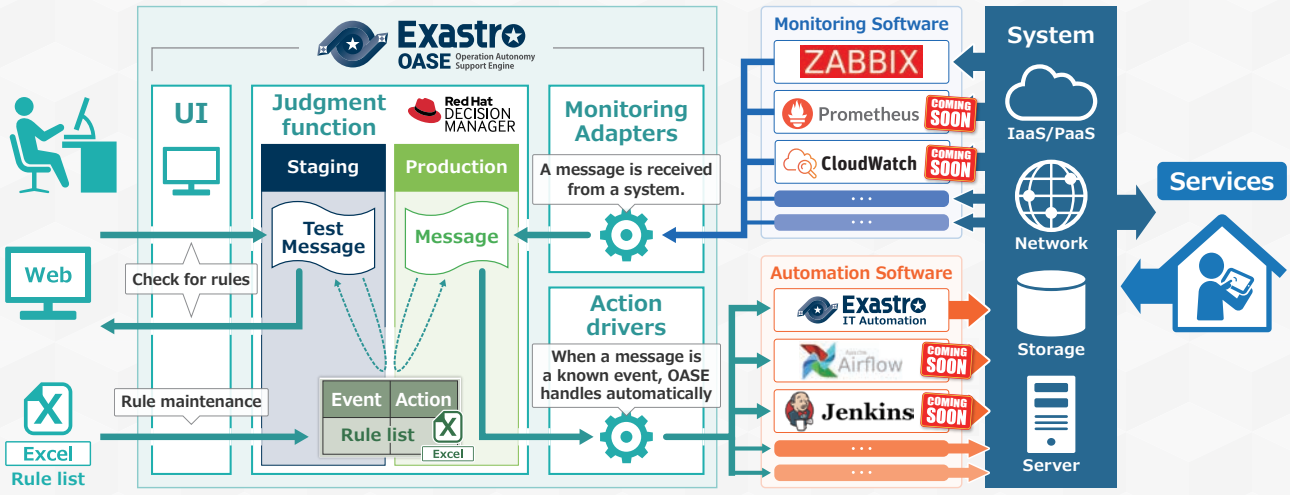
It can be a bit hard to automate all the steps in one sequence, so we recommend automating all the steps up until step 4 (Investigation) into one sequence and then partly automating step 5 (Fundamental countermeasures).

Supporting the actualization of Defensive Automation.

「Exastro Operation Autonomy Support Engine」(Exastro OASE)

The open source software, Exastro OASE, is an extremely effective tool for handling said obstacle countermeasures.

Exastro OASE links monitoring software and automation software together and can tell known and unknown problems apart from each other. If the obstacle or problem is already known, it automatically decides what kind of countermeasure to take.



The **6 features** of Exastro OASE

1. Confirm rule list in staging environment
2. Link with multiple monitoring software and automation software
3. Define the rules without coding
4. Control the number of action execution
5. Manage multiple rule list
6. Role-based access control

In this section, we will explain the features: "Define the rules without coding" and "Manage multiple rule list".

Define the rules without coding

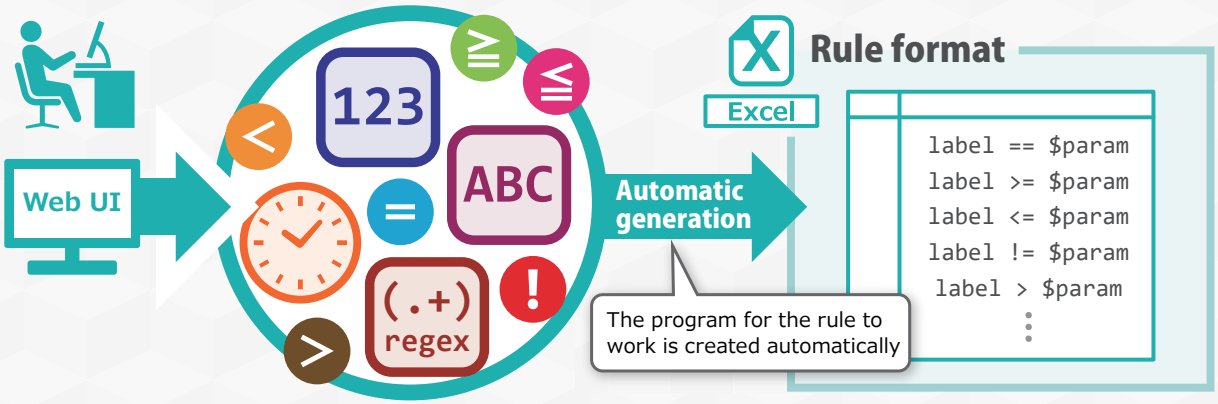
There might be some people who thinks that it might be possible to do temporary countermeasures in the monitoring software when a problem occurs. Indeed, Software that are able to register countermeasures and executing automatically do exists, such as Zabbix. However, functions like that often have be coded and have the contents of the countermeasures described to them, making it a big hurdle for users who usually don’ t code.

That is where by introducing "Red Hat Decision Manager" to the system in Exastro OASE comes in. One can define countermeasures to problem messages, meaning that it is possible to define rules in Excel.

Excel is widely used by many people for several different types of operations, making it easy for many to use.

While using Excel requires users to have pre-coded definitions prepared, Exastro OASE can do all the necessary coding automatically.

Because of that, the operators can write "Conditions for executing countermeasures" and "Countermeasure method" in their natural language (It is also possible to use Regular Expressions).



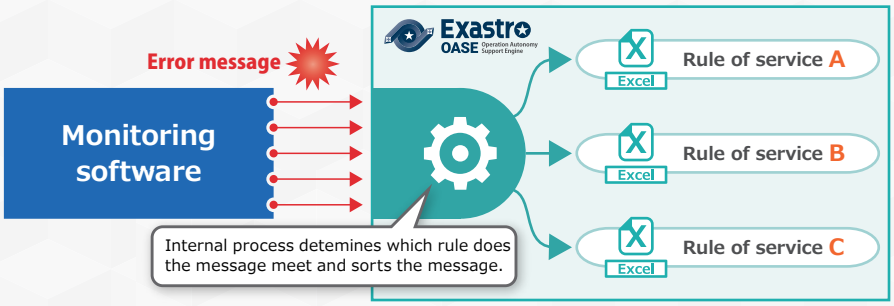
Manage multiple rule list

Exastro OASE allows users to create multiple rule lists.

As a result, by for example managing service units, Tthe granularity of management can be configured according to the organizational structure of the user.

As appearing error messages can set rule lists, they will be divided automatically. Therefore, there will be no need to worry about other rule lists.

Additionally, since access controll can be set per rule list, it is possible to maintain separate administrators.



Actualizing Defensive Automation will make Offensive Automation a weapon.

「Exastro IT Automation」(Exastro ITA)

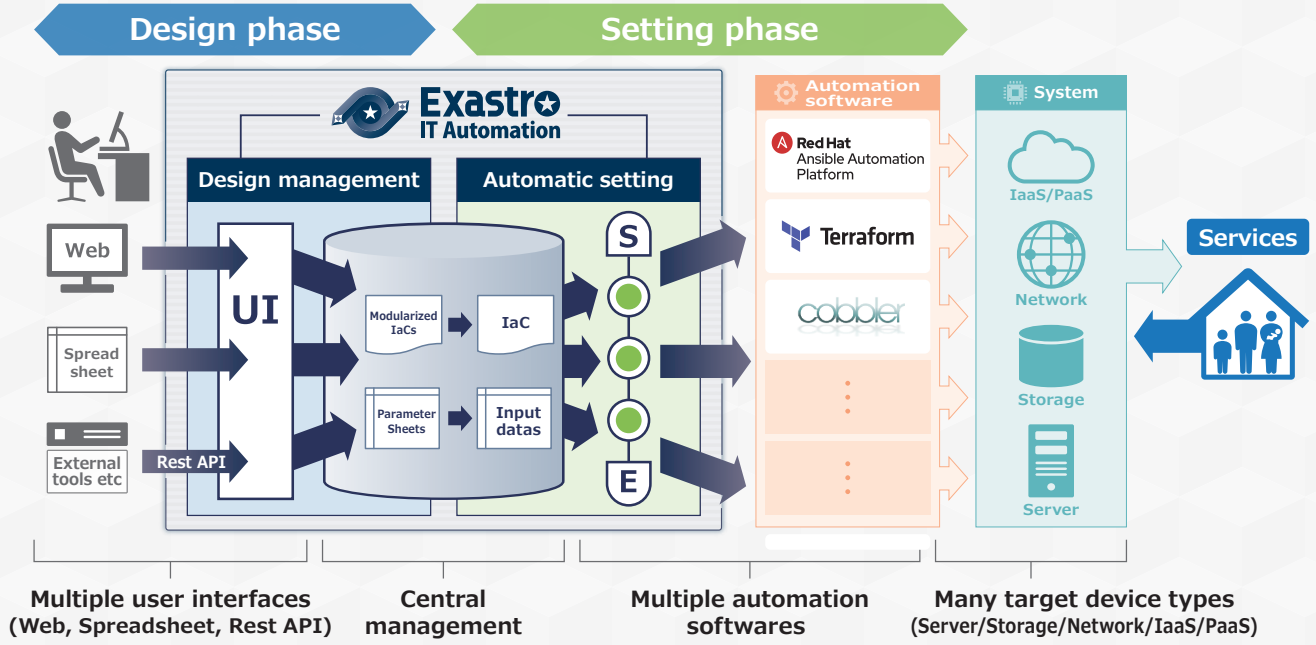
Exastro ITA, another open source software, is an useful tool for executing operations.

Exastro ITA can centrally manage the necessary IaC (Infrastructure as Code) and the parameter information needed to execute them.

If linked together with an automation software, it can also automatically execute them.

By managing the execution of Automation softwares, such as Red Hat Ansible Automation Platform(Ansible) and Terraform Enterprise(Terraform), which not only handle traditional monolithic systems, but also cloud-native systems.

As such, it is a software capable of handling both offensive and defensive automation.



The **7 Functions** of Exastro ITA

1. Multiple user Interfaces and RBAC (Role Base Access Control)
2. Manage parameters' change history
3. Prevent typos of variable names
4. Improve the reusability of the IaC
5. Control multiple automation tools
6. The last way to continue automation: pioneer mode
7. Real-time monitoring of execution status

In this section, we will explain the "Function to improve the reusability of the IaC" and “Function to control multiple automation tools”

Function to improve the reusability of the IaC

Only using IaC (Playbooks and such) once would be wasteful. If possible, IaCs should be reused as much as possible, making maintenance easier.

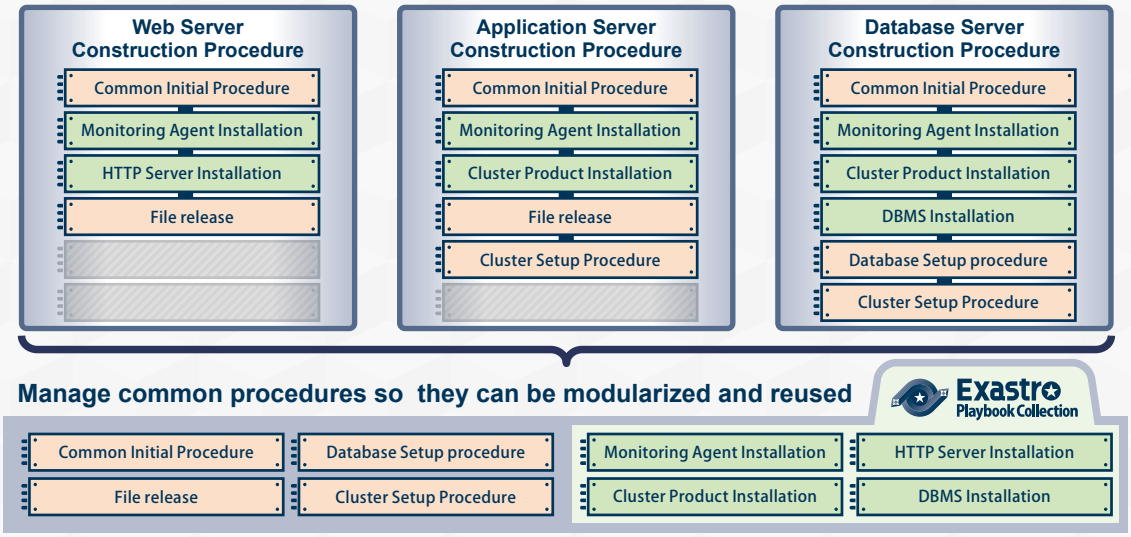
In Exastro ITA, users can make IaCs into modules, therefore increasing their reusability while being constructed during an operation.

Let`s take the figure below as an example. The 3 construction procedures might look completely different at first glance, but what if we make them into modules?

We can now see that the first construction step, File release, Cluster settings and more are the same.

By making them into modules in advance and managing them as such, we can use the same module multiple times.

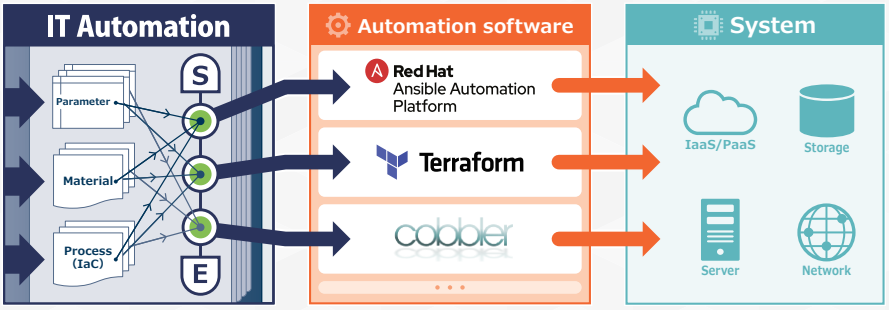
As a result of managing the modulated IaC in Exastro ITA, executing becomes easier.



Function to control multiple automation tools.

Exastro ITA isn`t just a software that automatically executes and manages IaC and Parameters. In order to operate automation software, it is necessary to create the information said software requires. Exastro ITA can do that automatically.

For example, in Ansible, it is necessary to create host_vars for every node that is getting executed. However, as Exastro ITA creates them automatically, there is no need to worry about them. With the use of it, executing tenfold, if not hundreds of nodes becomes easier.



Exastro community site