Force

Force is basically an event processing framework that emphasises on process automation for improvement of the operations. It is an infrastructure automation tool. Through Force we can create functions for the automatic processing of real-time events and extracting information from event streams as they arrive.

Force helps to create customisable event driven workflows.

It is helpful whenever we want a task to be redone after meeting a specific condition of the event.

It has the ability to write .force scripts to create event-driven workflows with multiple services combined together: Github to Docker builds, Slack to Kubernetes Deployments.

Force was developed with the ideology that there should be a high level abstraction, a glue language to describe modern cloud-native workloads.

Force being an interpreted mix of Go and Scheme makes use of Go’s CSP design to make it fun to work with concurrently running processes along with using Scheme’s functional declarative style that works well for cloud-native workloads where state is propagated across distributed infrastructure making it an excellent choice for creating event-driven workflows.

The main objectives of force -

It should be easy and fun to define declarative event-driven workflows for infrastructure projects.

* The tooling will be tailored to detect loops, inefficiencies in event-driven workflows.
* Should make it easy and manageable to have an even driven distributed system running on Kubernetes or on developer's laptop.
* Should be a single binary with no external dependencies.
* Should not invent a new syntax and use Go syntax for everything.
* It should be trivial to build a simple CI/CD system for a small project.

## **Batteries included**

Force already includes out of the box plugins for:

* Local and Kubernetes-native linux Docker builds.
* Github and git integration
* AWS S3
* SSH
* Slack

Soon force will include out of the box plugins for:

* Better integration with Kubernetes.
* Other popular source control and code sharing systems - Bitbucket, Gitlab.
* Event queues - Redis, Kafka, AWS SQS

The force repository contains various files that help us understand the basic ability of Force , such as

To showcase force's ability to create event driven workflows, let's create ticker.force that will print a tick every second:

Source file: ./docs/snippets/ticker.force

0: *// Tick every second and print current time*

1: Process(Spec{

2: Watch: Ticker("1s"),

3: Run: Printf("Tick!\n"),

4: })

The script in ticker.force starts with a Process(Spec{}) section. Watch subscribes to event channel, in this case Ticker that generates events.

Every time the event is generated, one, or several Run actions are triggered:

*# Press Ctrl-C to stop the script.*

$ force ticker.force

Tick!

Tick!

How CI / CD helps in automation -

In case you don’t know about it, CI / CD (Continuous integration and continuous delivery/deployment) are the concepts that allow the code to be accessible to more quickly and safely in order to generate a value.

Whenever some code is available on our local machine on some editor, we need to compile and execute each file to check the test cases, which makes testing of a framework very difficult. CI/CD contains tools helps to execute a complete framework.

GitHub is a major example using CI/CD which believes in build and automation testing.

It uses various execution tools like Jenkins and delivery tools like Docker.

How is force different from ci cd-

CICD is different from force as data integration and delivery places all the data at one place and output is generated as required , while in force , parameters are set according to the event so that a specific task must be performed.

CICD is implicit to Force .CICD pipelines can be used to implement Force as we think about the processes that could be automated whenever any change is introduced.

In modern technology, clients need continuous updates CI/CD pipelines help to keep the updates flowing and Force helps it reach to the client by generating specific messages.

Also in CI/CD we are required to give continuous feedback which can be done through Force. We can simply write the reviews whenever a successful execution takes place.

Force-

It uses various services like github, dockers slack and kubernetes deployments.

It is stand alone framework that can be imported by any project or distributed applications.

Instead of using any complex functionality , force uses .force file for the execution of commands.

Understanding Force through Amazon SWF -

SWF-

Amazon swf(simple workflow service) is an amazon web service tool.

It is basically a state tracker system which keeps track of all the tasks being performed in a cloud or simply the workflow to coordinate work across distributed components. Due to which it is known as a cloud workflow web service. It later fetches the functions from the big data, in account of the scenario based on real-time event.

Force has the same motive as of SWF but differ in functionality and usages.

In swf we need to create a development environment appropriate to the programming language being used .So we will need to install various sdks whereas fore is a single binary file including the plugins being used.

AWS uses asynchronous workflow.

SWF does not execute any code.

SWF does not contain the logic of the workflow.

Kubernetes vs Docker

Docker is a containerization platform, and Kubernetes is a container orchestrator for container platforms like Docker.

“Kubernetes” is basically referred as a control loop for an entire container based environment . In reality, they are not directly comparable, have different roots, and solve for different things.

Docker is a platform and tool for building, distributing, and running Docker containers. It offers its own native clustering tool that can be used to orchestrate and schedule containers on machine clusters. Kubernetes is a container orchestration system for Docker containers .It works around the concept of pods, which are scheduling units (and can contain one or more containers) in the Kubernetes ecosystem, and they are distributed among nodes to provide high availability. One can easily run a Docker build on a Kubernetes cluster.