**CLOUDATE**

**Synopsis**

**Objective:**

To design a scripting tool that will automate the deployment of new releases of software, (thereby reducing time as seen in the current situation) to faster execution time from the current scenario.

**Problem Statement:**

The problem has been extracted from a class of movement or practices known as DevOps, which deals with automation of the process of software delivery and infrastructure changes.

The developer has to deploy the new releases of software using a tool called winSCP which transfers the updated codes remotely to the ec2-instance at a very low transfer rate. During this process the developer is unable to perform other tasks. Also the service provided by earlier version to the end user has to stop as well making it extremely in convenient for the developer. In the meanwhile, the service provided by early version may continue to provide live data which is ignored. The problem is aggravated when the developer has more than one ec2 instance to work on.

**Project Analysis:**

**Phase 1: (Plan, Learning and Assumptions)**

The plan of the project was to create a tool for updating the code which is deployed in the cloud instance that can be rolled-back if needed. In this case the cloud instance is AWS ec2.

To design the same, first challenge faced was to learn a scripting language that can be used to work with ec2 instances and other AWS tools. Python with its various tools and libraries was found to be most appropriate to script the proposed automation.

After selection of our scripting language we started online research of how initially the developer updates their code. On doing so I studied about the version control system specifically ‘git’. The next phase was to learn about the Amazon web Services. With the help of my project guide and various tutorials I learned to work with AWS ec2 instance and S3 bucket.

**Phase 2: (Creating two versions)**

After learning various tools and technology, the scripting was commenced. The first thing I needed was two version of software to check how update works. I created two version of a simple script. On doing so I also learned how python packaging system works. So now I can manually update my python library with the help of pip. I chose pip for automating the above process. In this process an assumption is made that the update should be isolated. On doing so I found the tool virtualenv which separates environment from one another. I assumed we can use this for rolling back the update.

**Phase 3: (Assumption made was false)**

The use of virtualenv as to revert back the changes is invalid assumption taken as the use of virtualenv and pip has limitation over the python modules. There may be a case where the project has non python codes.

To overcome this problem we decided to run a local git where backup is to be created for reverting back the changes.

Another assumption is made that the script can be run remotely without logging into the instance with putty. To do so I learned a tool known as PARAMIKO. This helps us to remotely send commands to server. But the domain of the tool is vast and extensive. At last remote logging is kept aside and proceeded as to run the script into the remote server with putty.

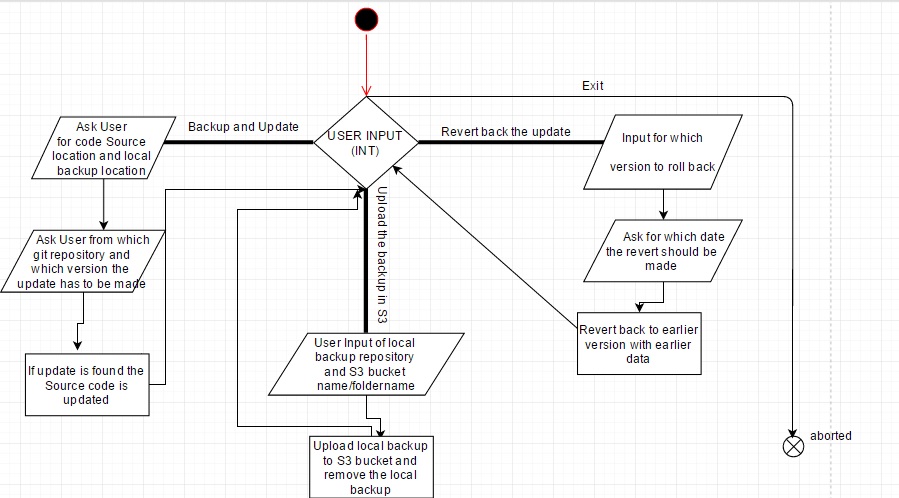
**Phase 4: (Final stage as v1.1)**

At last final tool has been created with the use of git for version controlling, awscli for saving the backup into the S3 bucket. In this phase we can backup our code without the pip and python virtualenv dependencies.

The script developed can be used as semi automated update of software code running on ec2 instance with saving earlier version for revert back from S3 bucket.

Prerequisite:

1. Git
2. AWS ec2 and S3 services
3. Python (for running the script)
4. Virtualenv and pip (for python codes)

**Workflow:**

**Team:**

DeveloperConcept and Guide

Abhishek Das Ranjit Sankar

Sarci Automation Sarci Automation

Intern Director

**References:**

Python Tutorials

* <http://www.learnpython.org/>
* <https://www.youtube.com/channel/UCfzlCWGWYyIQ0aLC5w48gBQ>
* <https://docs.python.org/2/whatsnew/2.7.html>
* Stackoverflow.com

AWS Tutorials

* Basic knowledge provided by project guide
* <http://docs.aws.amazon.com/gettingstarted/latest/awsgsg-intro/gsg-aws-tutorials.html>
* <https://aws.amazon.com/articles/3998>

Boto and Paramiko

* <http://boto.cloudhackers.com/en/latest/>
* https://www.youtube.com/watch?v=A7F3s0DdhZA