Adaptation in Cloud computing

Term Paper Rohit Kumar

1. Preliminaries

Cloud elasticity is defined as the degree to which a system is able to adapt to workload changes by provisioning and de-provisioning resources in an autonomic manner, such that at each point in time the available resources match the current demand as closely as possible. Through this, Software as a service (SaaS) provider relying on infrastruture as a service (IaaS) have the capability to quickly cope with highly and unpredictable demands by finely allocating resouces accordingly. Therefore meeting Service Level Agreements (SLAs) previously established with their customers.

Infrastructure Horizontal elasticity HS_{infra} basically refers to the on-demand adding of a new VM. This is applicable for applications that have clustered architecture.Clustered architecture is needed as a master node is needed that distributes requests between the worker nodes.Cost of this depands upon the ease with which nodes can join or leave the cluster.

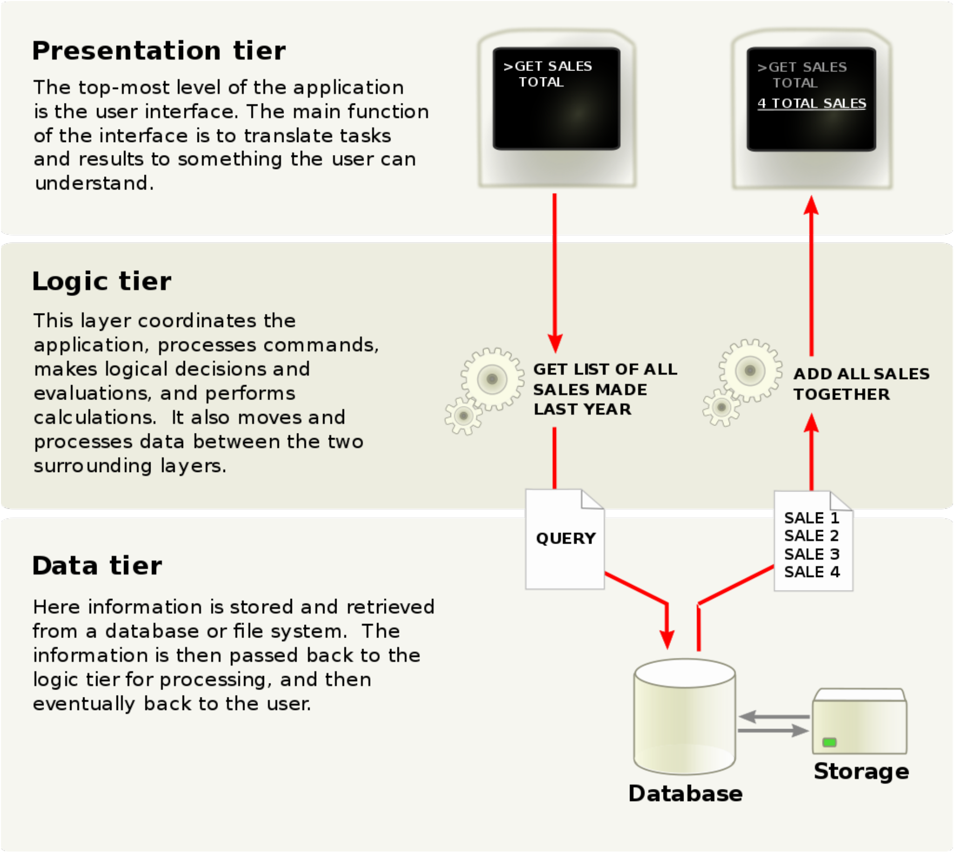
Infrastructure vertical elasticity VS_{infra} refers to the adding / deleting RAM or CPU to/from the VM system. Modern hypervisiors support online VM resizing allowing one to add CPU or memory resources to a VM without bringing it down. It is limited by the amount of free CPU Cores and memory available on the physical server hosting the virtual machine. Modern hypervisior also support **live migration** allowing VM to be migrated from one physical server to another.

Software Elasticity is the capability of a software to adapt itself to meet demand changes and/ or mitigate the infrastruture resource limitiations. It can be done in software in two ways : remove software components on the fly – HS_{Soft} and change the offering of existing components VS_{soft} .

Multi-tier software model

It is a modelling principle where software is composed of multi – tiers, thus make each tier independent of other in terms of resoruces. Three layer model of it is the most common.

This helps us in using cloud flexibility. For e.g : Google app engine can be used for logic tier / layer and Amazon database service for data layer for the same software.



Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

Difference b/w elasticity mode and elasticity method

Elasticity mode is the needed interactions or manner in order to perform elasticiy actions. It handles the problem of how the need for adaptation is calculated. There are various divisons in it :

- Reactive mode
  - Static threshold
  - Dynamic threshold
- Proactive mode
  - Time series analysis
  - model solving mechanisms
- Model predictive control (MPC)
  - Reinforcement learning (RL)
  - Control theory

Elasticity methods

Once elasticity policy have decided for how much resources are needed, method handles how demanded resource is provided.

To deploy elasticity solutions, one or hybrid of the following methods is implemented to meet the demand.

- Infrastructure
    - Horizontal scaling
    - vertical Scaling
- Software level
    - Software elasticity

2. Problem

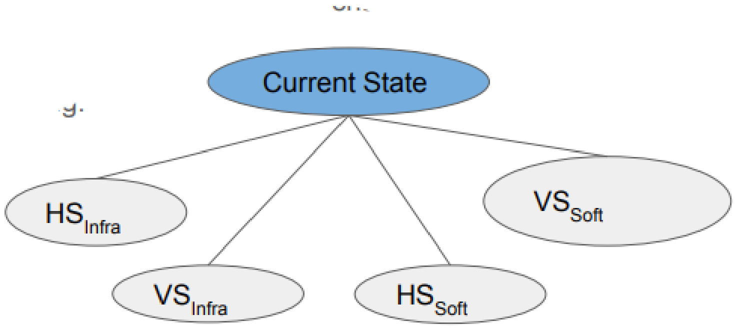
The cloud supporting online provisioning / deprovisioning of resources provides a flexible system of resources for the softwares. It helps in robust handling of dynamic traffic that service provider face from day to day.

Since the cloud offers us choice on how to handle the ongoing demand for the resources, we need to select optimal choice of resource such that it satisfies our needs and the cost of it is minimized, as generally we are paying for it.

It should be optimal in the sense that it handles the following recognised challenges :

- ✔ Resources are limited
- ✔ Resource initiation time can be too long
- ✔ Partial usage waste.

The choice can be represented by following digram.



3. Literature Survey

I reviewed the following three research papers.

1. How to Adapt Applications for the Cloud Environment

Summary : (since this paper is a review of the field, only a brief summary is mentioned)

- Presents a review of field.
- Three layer software development model as base and studies the adaptation layerwise.
  - Data layer
  - Business layer
  - Presentation layer
- Addresses the research questions and open challenges with respect to adaptation of each layer.

2. SmartScale: Automatic Application Scaling in Enterprise Clouds

Summary (we have already discussed the same paper in the class)

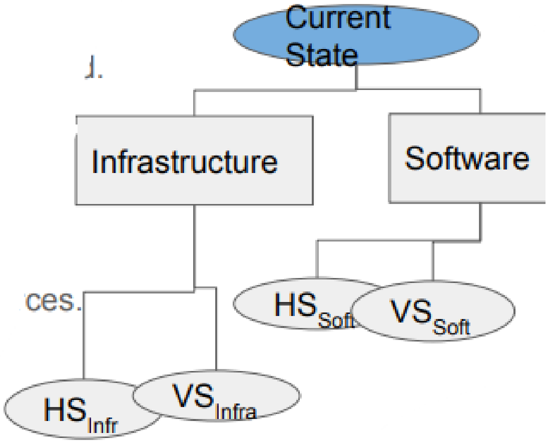
- Addresses the choice of horizontal scaling and vertical scaling as an optimisation problem
- Optimisation to minimize the joint cost of horizontal scaling and vertical scaling
- Uses binary search to find the minima.

3. Experimental Analysis on Autonomic Strategies for Cloud Elasticity

There are two types of resources at out hand that can be used to fullfill the incoming demand for more resources. Adding more resoruces from infrastruture or handling using the software elasticity. Each of the choice presents their own issues and benefits.

Infrastructure resoruces are limited , initiation time is significant to impact the service, and pricing is per instance hour.

Software elasticity have the inherent need to make the software modular and various component for the same feature by the order of weight. *However, the initiation time in SaaS is neglible compared to IaaS.*



Benefits of complementary usage of IaaS and SaaS elasticities

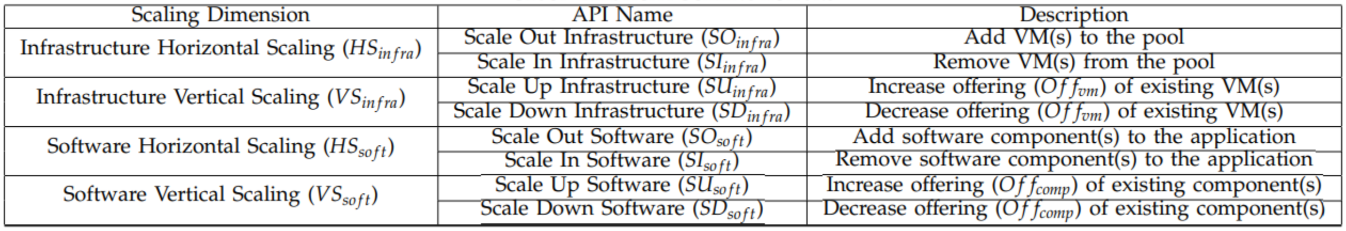
The very important impact of this is – SaaS elasticities have neglible initiation time while IaaS have significant initiation time. Thus during the time the IaaS resources are being prepared, the SaaS elasticities can handle the resource demand and thus mange to response to the traffic spike.

Others are :

- Alleviate the use of infrastruture resource
- improve responsiveness of scaling
- improve expression capabilites of elasticity.

Summary of possible elasticities and related actions.

Table I
Cloud Elasticity Scaling actions



Scaling Dimension	API Name	Description
Infrastructure Horizontal Scaling (HS_{infra})	Scale Out Infrastructure (SO_{infra})	Add VM(s) to the pool
	Scale In Infrastructure (SI_{infra})	Remove VM(s) from the pool
Infrastructure Vertical Scaling (VS_{infra})	Scale Up Infrastructure (SU_{infra})	Increase offering (Off_{vm}) of existing VM(s)
	Scale Down Infrastructure (SD_{infra})	Decrease offering (Off_{vm}) of existing VM(s)
Software Horizontal Scaling (HS_{soft})	Scale Out Software (SO_{soft})	Add software component(s) to the application
	Scale In Software (SI_{soft})	Remove software component(s) to the application
Software Vertical Scaling (VS_{soft})	Scale Up Software (SU_{soft})	Increase offering (Off_{comp}) of existing component(s)
	Scale Down Software (SD_{soft})	Decrease offering (Off_{comp}) of existing component(s)

Paper approach

Model Introduction

The paper formulates a autoscaling service that uses the previously defined elasticities to handle the resource demand. The model presented is based on well known MAPE-K formalism. The interacts with the managed system , the cloud resources graph using sensors and actuators, thus taking the resource management to a new level of abstaction, separate from the resource system itself.

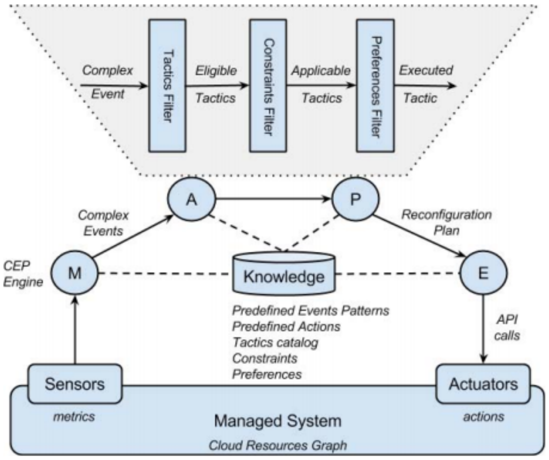


Figure 4. Cloud Resources Model Overview

Managed system

The managed system taken is a n-tier web applications. Each tier is hosted on a VM and VM’s are hosted on a set of physical machines.

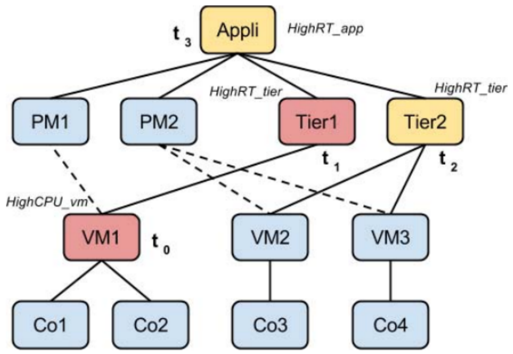


Figure 5. Instance of cloud resources model (managed system)

Monitor : Events

Each cloud resource has a set of metrics indicating it’s health. The monitor phase collect and aggregate multiple sensor values over time. This helps system ot make timely relevant information about the system health in form of **events**. The event can be basic and complex, wich basuc representing the event handling resource management of a single resource. Complex are formed as a composition of basic events.

CA (cloud administrator) , person incharge of the cloud management, needs to detect complex relationships b/w events overtime by definining rules also known as **event patterns.**

Execute : predefined actions

There are a set of predefined actions defined by CA. The paper presents two types of actions : Basic actions and complex actions. The goal of complex actions is to absorb the infrastruture resoruce initiation time.

An example of complex action is shown in the figure below. We got a request / need for more resource, we do two things : we scale out infrastruture resouces but since the initiation time will be large, so what we do is we use software elasticity – making software light , till the infrastruture resouces are initialised. Than we normalise the software to the normal weight. This is one complex action.

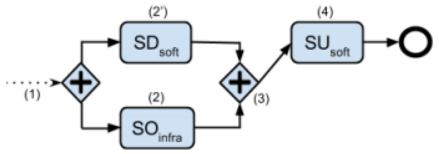
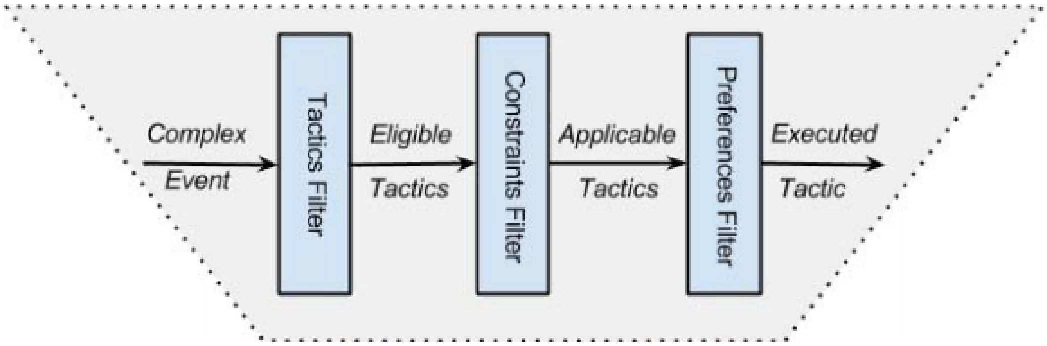


Figure 6. Complex action example: $(SD_{soft} \parallel SO_{infra})$; SU_{soft}

Analyse and Plan : Reconfiguration decision

We have seen the M and E of MAPE. Bridging the gap b/w M and E is the AP. The author presented this step as three step decision making.



- **Tactic filter –** A tactic is acutally an event-action pair defined by CA and represents a well-known IF-THEN statement. The step 1 of 3 looks for tactics corresponding to the event stored in Knowledge by CA.
- **Constraints Filter –** The runtime context can induce some constraints that may prevent to execute eligible tactics.
- **Preferences filter –** This filter a final complex action based on CA preferences. Some examples of preferences are : Cost, QoS, QoE, Elasticity time, Responsiveness.

Knowledge

It is the data shared among all MAPE phases such as current cloud resources graph with associated metric values, runtime constraints , preferences expressed on the system etc. There are other defined by CA that are crucial for decision making like event patterns, predefined actions, tactics.

4. Additional papers

A Survey on Cloud Computing Elasticity - Link

1. Elasticity in Cloud Computing: What It Is, and What It Is Not - Link

2. Horizontal and Vertical Scaling of Container-Based Applications Using Reinforcement Learning - Link
3. Elasticity in Cloud Computing: State of the Art and Research Challenges - Link

4. Preferred Approach

SmartScale does not take into account the infrastruture limitations into account which are very crucial for a system working in real-time. Handling the limitations using cross-layered elasticity is one of the possible solutions to address the limitiations.

5. My ideas on possible extensions

In the paper we discussed, there is no optimisation of infrastructure resource selection. I.e which one to select – Vertical or Horizontal scaling. Which one is optimal choice ? **The first extensions is to include the optimisation of resouce selection.**

The second extension that is very clear form summary itself - The involment of CA is quite huge, nearly all decision making is done by forming complex event and complex actions based on the CA knowledge from trial runs. There is inherent need in the paper for iterative improving by CA.

We need to minimise the human involment in the decision making. One possible way to do so by using RL with human in loop for feedback. This makes it possible to improve the decision making and keep in check with human feedback also. I propose human feedback because mathmatical formalism of reward in such cases may be not directly feasible. It will be good if system is able to formulate a reward / penalty mathmatically which remove the depandancy on the human feedback which brings subjective issue into the frame.