ISSN:0975-9646

Vishweshwaraiah.L.M et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (4) , 2014, 5726-5728

The Migration to Meta Cloud from Vendor Lock-In

Vishweshwaraiah.L.M[1], N.Parashuram[2], Dr.S.Prem kumar[3]

\

[1] *G.Pullaiah College Of Engineering,M.Tech*
Kurnool, India
[2]*G.Pullaiah College Of Engineering,M.Tech, Asst. prof*
Kurnool, India
[3]*G.Pullaiah College Of Engineering,M.Tech, M.phil, PH.D, HOD*
Kurnool, India

Abstract:Cloud computing has proved efficient in recent years because of the customer's ability to use services on demand with a pay-as-you go pricing model which has proved convenient in each and every aspect. Low costs and high flexibility make migrating to cloud compelling. Despite its obvious advantages, however, many companies hesitate to "move to the cloud," mainly because of concerns related to service availability, data lock-in, and legal uncertainties.1 Lock in is particularly problematic, even though the public cloud availability is very high, outages still occur.2 Business locked into such a cloud is particularly in the locked state until the cloud back in online. Moreover this public cloud providers generally don't agree with Service level agreements(SLA's).3 that is business locked in such a cloud will not provide quality of service(QOS).Hence the business locked onto a cloud has no mid-long term control over its own IT costs.

In order to resolve all these problems I identified a need for businesses to permanently monitor the cloud they are using and be rapidly "change horses" which means that if the problem was found in one cloud then migrate to another cloud in order to resolve future issues.

Index terms: Cloud computing, Distributed computing, Meta-cloud.

INTRODUCTION:

Cloud computing is an emerging technology proved efficient in the recent years.Cloud computing has achieved a widespread adoption in the recent years. Its success is due mostly to customers' ability to use services on demand with a pay-as-you go pricing model, that has proved convenient in many respects. Low prices and high flexibility make migrating to the cloud compelling. Despite its obvious benefits, however, several firms hesitate to "move to the cloud," mainly because of issues associated with service convenience, data lock-in, and legal uncertainties.1 Lockin is particularly problematic. For one issue, even though public cloud convenience is usually high, outages still occur.2 Businesses barred into such a cloud square measure primarily at a standstill until the cloud is back on-line. Moreover, public cloud suppliers usually don't guarantee particular service level agreements (SLAs)3 that's, businesses barred into a cloud don't have any guarantees that it'll still give the specified quality of service (QoS). Finally, most public cloud suppliers' terms of service let that provider unilaterally amendment valuation at any time. Hence, a business barred into a cloud has no mid- or long term control over its own IT prices.

At the core of of these issues, we can identify a necessity for businesses to for good monitor the cloud they're victimisation and be able to rapidly "change horses" — that's, migrate to a different cloud if they discover issues or if their estimates predict future problems. However, migration is presently off from trivial. Myriad cloud providers square measure f looding the market with a confusing body of services, together with work out services like the Amazon Elastic work out Cloud (EC2) and VMware vCloud, or key-value stores, such as the Amazon straightforward Storage Service (S3). a number of these services square measure conceptually comparable to one another, whereas others square measure vastly totally different, however they're all, ultimately, technical ly incompat ible and follow no standards however their own. To additional complicate the scenario, several firms not (only) build on public clouds for his or her cloud computing needs, however mix public offerings with their own personal clouds, resulting in questionable hybrid cloud setups.4 Here, we tend to introduce the thought of a meta cloud that incorporates style time and runtime parts. This meta cloud would abstract away from existing offerings' technical incompatibilities, thus mitigating trafficker lock-in. It helps users realize the correct set of cloud services for a particular use case and supports an application's initial readying and runtime migration.

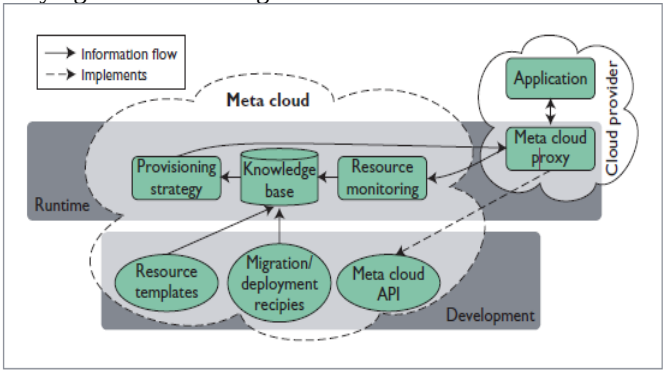

Figure 1. Conceptual meta cloud overview. Developers create cloud applications using meta cloud development components. The meta cloud runtime abstracts from provider specifics using proxy objects, and automates application life-cycle management.

METACLOUD:

The meta cloud API provides a unified programming interface to abstract from the variations among supplier API implementations. for purchasers, using this API

prevents their application from being hard-wired to a particular cloud service giving. The meta cloud API will repose on available cloud supplier abstraction APIs, as antecedently mentioned. Although these deal largely with keyvalue stores and work out services, in principle, all services will be lined that area unit abstract enough for a lot of than one supplier to supply and whose specific arthropod genus don't take issue an excessive amount of, conceptually.

RESOURCE TEMPLATES:

Developers describe the cloud services necessary to run an application using resource templates. They can specify service sorts with extra proper ties, and a graph model expresses the relation and functional dependencies between services. Developers produce the meta cloud resource templates employing a simple domain-specific language (DSL), property them shortly specify required resources. Resource definitions are supported a ranked composition model; so developers can produce configurable and reusable template elements, that modify them and their groups to share and reuse common resource templates in different comes. victimization the phone line, developers model their application components and their basic runtime requirements, like (provider independently normalized) computer hardware, memory, and I/O capacities, as well as dependencies and weighted communication relations between these elements. The provisioning strategy uses the weighted part relations to work out the application's best readying configuration. Moreover, resource templates enable developers to outline constraints supported prices, component proximity, and geographical distribution.

MIGRATION AND DEPLOYMENT RECIPES:

Deployment recipes square measure a crucial ingredient for automation within the meta cloud infrastructure. Such recipes allow for controlled preparation of the application, as well as putting in packages, beginning needed services, managing package and application parameters, and establishing links between connected parts. Automation tools like Opscode Chef offer an in depth set of functionalities that square measure directly integrated into the meta cloud surroundings. Migration recipes go one step further and describe the way to migrate an application throughout runtime for example, migrate storage practicality from one service supplier to another. Recipes solely describe initial deployment and migration; the provisioning strategy and also the meta cloud proxy execute the particular method using the said automation tools.

META CLOUD PROXY:

The meta cloud provides proxy objects, that are deployed with the application and run on the provisioned cloud resources. They serve as mediators between the appliance and the cloud supplier. These proxies expose the meta cloud API to the application, remodel application requests into cloud-provider-specific requests, and forward them to the respective cloud services. Proxies provide some way to execute readying and migration recipes triggered by the meta cloud's provisioning strategy.Moreover, proxy objects send QoS statistics to the resource watching component running among the meta cloud. The meta cloud obtains the data by intercepting the application's calls to the underlying cloud services and activity their process time, or by capital punishment short benchmark programs. Applications can even outline and monitor custom QoS metrics that the proxy objects send to the resource watching element to enable advanced, application-specific management ways. To avoid high load and machine bottlenecks, communication between proxies and the meta cloud is unbroken at a minimum. Proxies don't run within the meta cloud, and regular service calls from the application to the proxy aren't routed through the meta cloud, either.

RESOURCE MONITORING:

On associate application's request, the resource monitoring element receives knowledge collected by meta cloud proxies regarding the resources they're using. The element filters and processes these knowledge so stores them on the mental object for more process. This helps generate comprehensive QoS data regarding cloud Figure 1. abstract meta cloud summary. Developers produce cloud applications mistreatment Meta cloud development parts. The meta cloud runtime abstracts from supplier specifics mistreatment proxy objects, and automates application life-cycle management. Runtime Information on how Implements service suppliers and therefore the explicit services they supply, including response time, handiness, and more service-specific quality statements.

PROVISIONING STRATEGY:

The provisioning strategy element primarily matches Associate in Nursing application's cloud service necessities to actual cloud service suppliers. It finds and ranks cloud services supported information in the content. The initial deployment call is predicated on the resource templates, specifying the resource necessities of Associate in Nursing application, together with QoS and evaluation information concerning service suppliers. The result's an inventory of doable cloud service mixtures graded according to expected QoS and prices. At runtime, the element will reason about whether or not migrating a resource to another resource supplier is helpful based on new insights into the application's behavior and updated cloud supplier QoS or evaluation information. Reasoning concerning migrating additionally involves calculative migration prices. Decisions concerning the provisioning strategy lead to the element death penalty customer-defined preparation or migration scripts.

KNOWLEDGE BASE

The content stores information concerning cloud supplier services, their evaluation and QoS, and data necessary to estimate migration prices. It additionally stores customer-provided resource templates and migration or deployment recipes. The data base indicates that cloud suppliers are eligible for a definite client.These sometimes comprise all suppliers the client has Associate in Nursing account with and providers that supply prospects for creating (sub)accounts on the fly. Several information sources contribute to the data base: meta cloud proxies regularly

send information concerning application behavior and cloud service QoS. Users will add cloud service providers' pricing and capabilities manually or use creeping techniques that may get this info mechanically.

META CLOUD A USE CASE:

For initial study, the developer submits the application's resource template to the meta cloud. It specifies not solely the 3 varieties of cloud services required to run the sports application, however conjointly their necessary properties and the way they rely on each other.

The developer will specify CPU, RAM, and space according to nomenclature outlined by the meta cloud resource example digital subscriber line. Each resource is named within the example, which permits for referencing throughout deployment, runtime, and migration. The resource example specification should conjointly contain interdependencies, such as the direct affiliation between the Web service figure instances and the message queue service. The made data that resource templates give helps the provisioning strategy element create profound selections regarding cloud service ranking. we are able to justify the working principle for initial study with an internet search analogy, in which resource templates area unit queries and cloud service supplier QoS and rating data represent indexed documents. Recursive aspects of the particular ranking area unit beyond this article's scope. If some resources within the resource graph area unit only loosely coupled, then the meta cloud are going to be additional doubtless to pick out resources from totally different cloud suppliers for one application. In our use case, however, we tend to assume that the provisioning strategy ranks the several Amazon cloud services initial, and that the client follows this recommendation. After the resources area unit determined, the meta cloud deploys the application, at the side of associate degree instance of the meta cloud proxy, according to customer-provided recipes. During runtime, the meta cloud proxy mediates between the appliance components and also the Amazon cloud resources and sends observation information to the resource observation element running at intervals the meta cloud. Monitoring information helps refine the application's resource example and the provider's overall QoS values, both keep within the mental object. The provisioning strategy element regularly checks this updated information, which could trigger a migration. The meta cloud may migrate front-end nodes to alternative providers to put them nearer to the application's users, as an example. Another reason for a migration might be updated rating information.To create these decisions, the provisioning strategy component should contemplate potential migration prices relating to time and money. the particular migration is performed based on customer-provided migration recipes. Working on the meta cloud, we face the subsequent technical challenges. Resource observation should collect and method information describing different cloud providers' services such that the provisioning strategy can compare and rank their QoS properties in a very normalized, provider independent fashion. though solutions for readying within the cloud area unit relatively mature, application migration isn't moreover supported. Finding the balance between migration facilities provided by the meta cloud.

Cloud-centric migration makes the meta cloud infrastructure accountable for most migration aspects, leading to problems with application specific intricacies, whereas in application-centric migration, the meta cloud solely triggers the migration process, exploit its execution mostly to the appliance. We argue that the meta cloud ought to management the migration method however supply several interception points for applications to influence the method in any respect stages.

The provisioning strategy the most integrative element, which derives methods in the main supported input from runtime observation and resource templates and effects them by capital punishment migration and readying recipes needs fur their research into combining approaches from the knowledge retrieval and autonomic computing fields. The meta cloud will facilitate mitigate vendor lock-in and guarantees transparent use of cloud computing services. Most of the essential technologies necessary to comprehend the meta cloud already exist, however lack integration. Thus, desegregation these progressive tools guarantees a large leap toward the meta cloud. To avoid meta cloud lock in, the community should drive the concepts and create a very open meta cloud with side worth for all customers and broad support for various suppliers and implementation technologies.

CONCLUSION:

The meta cloud which has been implemented can help mitigate vendor lock-in and promises transparent use of cloud computing services. Most of the recent technologies realize that meta cloud already exists but they are lack of integration. This integration problem in cloud will leads to business lock-in condition. This leads to meta cloud lock-in problem.To avoid meta cloud lock-in, the community must drive the ideas and create a truly open meta cloud. This will add the value for all customers, and will broad support for different providers and implementation technologies.

REFERENCES :

1. M. Armbrust et al., "A View of Cloud Computing," *Comm. ACM*, vol. 53, no. 4, 2010, pp. 50–58.
2. B.P. Rimal, E. Choi, and I. Lumb, "A Taxonomy and Survey of Cloud Computing Systems," *Proc. Int'l Conf. Networked Computing and Advanced Information Management*, IEEE CS Press, 2009, pp. 44–51.
3. J. Skene, D.D. Lamanna, and W. Emmerich, "Precise Service Level Agreements," *Proc. 26th Int'l Conf. Software Eng.* (ICSE 04), IEEE CS Press, 2004, pp. 179–188.
4. Q. Zhang, L. Cheng, and R. Boutaba, "Cloud Computing: State-of-the-Art and Research Challenges," *J. Internet Services and Applications*, vol. 1, no. 1, 2010, pp. 7–18.
5. M.D. Dikaiakos, A. Katsifodimos, and G. Pallis, "Minersoft: Software Retrieval in Grid and Cloud Computing Infrastructures," *ACM Trans. Internet Technology*, vol. 12, no. 1, 2012, pp. 2:1–2:34.