Azerbaijan Journal of High Performance Computing, Vol 3, Issue 1, 2020, pp. 15-31
https://doi.org/10.32010/26166127.2020.3.1.15.31

An Integrated Approach To Manage Iaas With Software-Defined Infrastructure (SDI) Management And Control System (MCS)

Muhammad Bayat[1], Hasan Hani[2]

[1]MajdRayan Intelligent Computing, Qom, Iran, m.bayat@qom.ac.ir
[2]University of Qom, Qom, Iran, hani@qom.ac.ir

*Correspondence:
Muhammad Bayat,
MajdRayan Intelligent
Computing, Qom, Iran,
m.bayat@qom.ac.ir

Abstract

Cloud computing is developing and spreading very quickly. One of the most important issues among cloud service providers is efficient resource allocation. The IaaS layer is responsible for providing the resources needed for cloud services. The automatic resource allocation to the workloads is another concern for cloud service providers. Current approaches in infra-structure as a service (IaaS) that use three separate resource management (compute, storage, and network resource management) are not capable of re-sponding to applications and multimedia services that need a guaranteed quality of services. Some applications and end-to-end quality of services depend on the performance of computing, network, and storage resources. Therefore these resources must be controlled and managed altogether. In this research, we first describe available architectures that integrate and manage heterogeneous resources. Then, we propose a software-defined infrastruc-ture (SDI) management and control system (MCS) architecture that incorpo-rates IaaS resource management based on the performance, integrated re-source management, support from heterogeneous sources, overhead reduc-tion, and automatic allocation of resources regarding the workload criteria. We use the Analytic Hierarchy Process (AHP) to analyze the architectures based on the mentioned criteria. We prepare pairwise comparison matrices for all major and minor criteria and use the feature sets of each architecture to fill these matrices. The results obtained from the AHP shows that the proposed architecture is of higher priority than others.

Keyword: Cloud computing, infrastructure as a Service (IaaS), Software-defined Networking(SDN), Software-defined Infrastructure(SDI), dynamic re-source allocation

1. Introduction

The businesses increasingly rely on the availability and efficiency of their IT infra-structure. The business operations also link to the agility and performance of the de-

ployment and continuous operation of IT. The users access their required services without paying attention to where they are and how to access them (Fox, A., Griffith, R., et al., 2009).

Infrastructure as a service (IaaS) can be considered as the most important part of cloud computing without which none of the other components are implemented and provided. This infrastructure includes servers, networks, storage devices, and operating system (Javan, M. S., & Akbari, M. K., 2011, November). IaaS offers physi-cal and virtual infrastructure services. It can be implemented in public, private, or hybrid mode.

In 2008, software-defined networking (SDN) was proposed by researchers from Berkley and Stanford universities proposed that software-defined networking (SDN) looked for adding software features to the network without any dependency and interference with hardware. SDN is the next generation of the network, decoupling the network's data and control plane. In other words, SDN has reduced dependency on hardware and software features to increase network intelligence (Lara, A., Kolasani, A., & Ramamurthy, B., 2013). With SDN, companies and organizations can pro-gram their network based on their requirements and proprietary capabilities. It uses a software-based controller to manage to forward the information of one or more switches. In the SDN networks, the hardware is only responsible for delivering the traffic according to the rule set by the controller. SDN controller can simplify the configuration and management of the network and can ease configuring security fea-tures. SDN also provides lots of capabilities such as software-based traffic analysis, centralized control over data traffic, controlling multiple switches from a single con-troller, dynamic updating of forwarding rules and flow abstraction, programmatic control of all forwarding operations, capabilities advertisement and statistics reporting (Bakshi, K., 2013).

The popularity of SDN encourages scientists to expand this concept for other aspects of a data center. The software-defined compute (SDC) can manage the computing functions through a central interface that sees all computing resources as one element. In SDC, the computational functions can happen in any number of hardware devices in the cloud, as needed, rather than be assigned to a specific hardware device (Quintero, D., Genovese, W. M. et al., 2015). Besides, the computing functions can be moved around to different pieces of virtual infrastructure, depending on the availability of the resources.

Software-defined storage (SDS) refers to data storage software to manage policy-based provisioning and management of data storage independent of the underlying hardware. SDS concentrates on storage virtualization to separate the storage hardware from the software that controls the storage infrastructure (Lara, A., Kolasani, A., & Ramamurthy, B., 2013; Li, C. S., Brech, B. L., Crowder, et al., 2014). It provides policy management for feature options such as deduplication, replication, thin provisioning, snapshots, and backup.

Current virtualization and cloud solutions allow only the necessary abstraction of computing, storage, and network resources in terms of their capacity. For simplifying

the abstraction of resources, these solutions only standardize their proprietary architecture. The management of heterogeneous resources that include computing, networking, and storage resources separately is not appropriate for applications and multimedia services that require guaranteed quality of services. In addition, current resource management is not capable of managing heterogeneous resources in combination with other resources such as programmable hardware, graphics processing units (GPUs), and network processors (Lin, T., Kang, J. M., Bannazadeh, H., & Leon-Garcia, A., 2014).

Figure 1 shows that in today's IaaS, computing, networking, and storage resources are managed and controlled separately with their own controllers. When a workload is defined, each controller based on its available resources does resource allocation. For instance, when a job is divided into multiple tasks, a compute controller allocates each task to a virtual machine (VM) with the most available resources. Compute con-troller is only aware of computing resources and does not have any knowledge about the network or storage resources. The network controller also controls and manages traffic flows regardless of other resources in the cloud.

There are two reasons why today's IaaS platforms fail to address the needs of users who run complex workloads. Firstly, workloads deployed in the cloud involve com-plex interconnections of virtual resources, such as VMs, virtual networks, virtual storages, and software components, and their connectivity is often left to the user. To automate the deployment of a workload, a cloud user must implement complex scripts. Secondly, performance, availability, and the cost of execution of a workload depend on how the IaaS platform maps virtual resources to physical infrastructure.

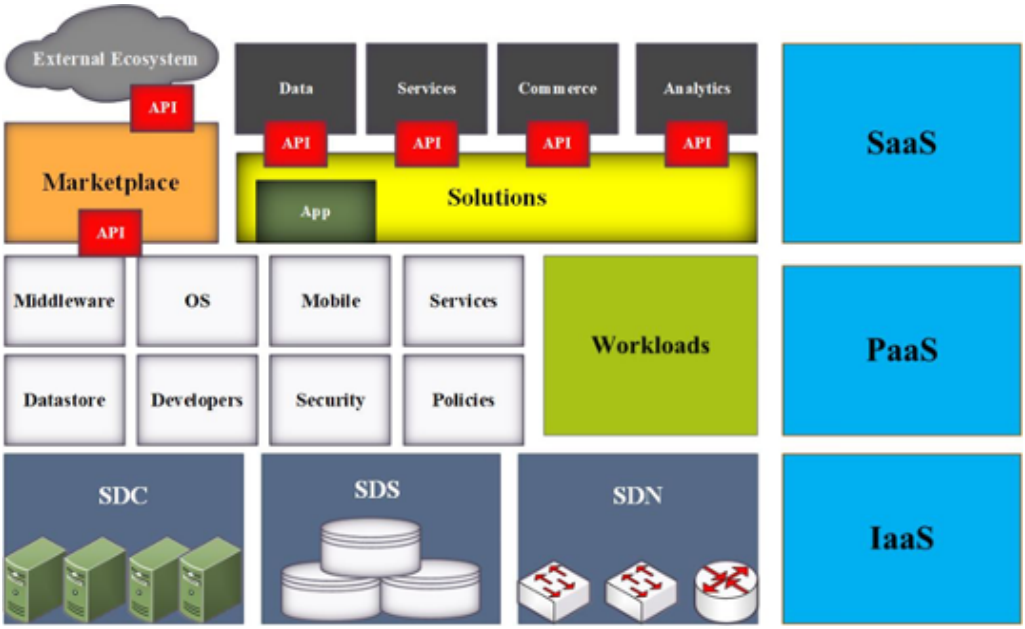

Fig. 1. Current cloud computing architecture ([8])

The paper is organized as follows. Section 2 describes related work to integrated resource management. The SDI resource management system (RMS) and software-defined environment (SDE) architectures will be described in this section. Section 3 presents SDI MCS as a new architecture for the infrastructure resource management system. In section 4, we evaluate and compare our architecture with the SDI RMS and SDE with the AHP. Finally, conclusions and future work are presented in section 5.

2. Related Work

2.1. SDI RMS

Thomas Lin et al. (2014) showed how they enabled SDN applications on software-defined infrastructure. To overcome the weakness of the current resource management, they recommend SDI as a software approach for integrated control and management of heterogeneous compute and network resources. They used the SAVI testbed to test their proposal. In SDI RMS, the SDI manager controls compute and network resources with the cloud and SDN controllers. Here, the pluggable resource management module is responsible for scheduling, network control, fault management, and so on. SDI RMS is based on the idea that all the network and computing resources can be virtualized. The principles of cloud computing are used in its infrastructure design to control and manage heterogeneous resources.

Figure 2 shows the high-level architecture of SDI RMS in which the SDI manager is responsible for controlling and managing networks and compute resources by using a cloud controller, an SDN controller, and a topology manager. External entities re-ceive virtual resources in the converged heterogeneous resources via SDI RMS. Con-verged heterogeneous resources refer to physical and virtual resources.

The SDI RMS prepares resource management functions to provide converged heterogeneous resources for the external entities. The external entities are applications, users, high-level management systems, and so on. The functions of resource management include virtualization, provisioning, migration, monitoring, security management, load balancing, and performance management.

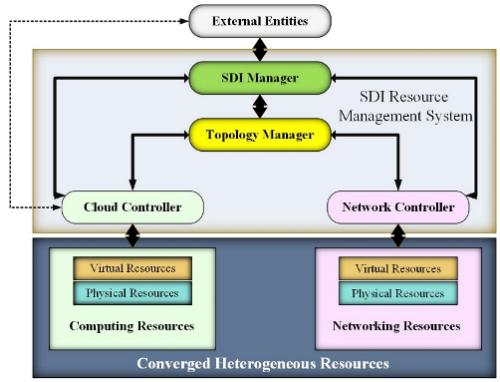

Fig. 2. High-level architecture of SDI RMS([7])

The SDI manager realizes network resource management functions such as fault tolerance, network-aware VM replacement, path optimization, quality of service (QoS), and real-time network monitoring based on the information it obtains via the topology manager. The cloud controller is responsible for computing resource man-agement, memory allocation, and VM placement. The SDN controller translates network specifications into high-level configuration commands and installed them on SDN-enabled networking resources.

The topology manager has a list of the resources and their relationships, prepares the latest physical and virtual network topology, and resource monitoring and measurement data to the SDI manager for topology-aware resource management. The SDI manager uses the topology manager for updating resource data. It uses the cloud controller for preparing computing resources. Similarly, the SDI manager uses the SDN controller for providing virtual network resources and monitoring data.

2.2. IBM SDE

In this section, we compare these architectures based on their feature set. Both sys-tems add very high capabilities to the IaaS. These capabilities include increasing effi-ciency and performance in IaaS besides preparing isolated virtual resources for upper cloud layers and multiple tenants. It should be mentioned that both systems use OpenStack for developing virtual infrastructure.

SDI RMS focuses on the network side of IaaS. In this model, the SDI controller is modular and uses the cloud controller, SDN controller, and topology manager to handle end-to-end packets. SDI RMS increases QoS for delay-sensitive traffic. It is also scalable, and it can handle growing amounts of packet-in requests in a graceful manner. The network control module in SDI RMS can completely process any num-ber of requests from all running virtual machines and computing resources.

The main problem of the SDI RMS is the lack of consideration of the storage resources. The single point of failure is the next drawback of this system. In SDI RMS, the SDI manager makes all of the decisions, and if the SDI manager fails, the whole system will be in trouble. The SDN controller, the cloud controller, and the topology manager send all the infrastructure events to the SDI manager to decide how to handle the traffic. However, if the SDI manager is in trouble, the quality of service requirements for critical traffic cannot be met. This system also lacks automatic resource allocation to the workloads. Workload orchestration and management is a significant part of the integrated resource management system, and without this, optimal resource allocation cannot be met. This system also does not provide an appropriate dashboard for monitoring allocated and unallocated virtual infrastructure resources. This dashboard is important for an integrated resource management system administrator who should define workload and infrastructure patterns and policies.

SDE tries to integrate all the infrastructure resources and offers a comprehensive ar-chitecture. The SDE uses patterns and policies for workloads and infrastructure resources to orchestrate and automate resource allocation to the workloads on demands.

SDI RMS lacks this capability and does not consider workloads patterns and policies in its resource allocation. Resource abstraction and workload pattern definition and orchestration are the most important requirements for cloud IaaS. It also lacks a topology management module. The topology manager, in the integrated resource management system, should keep a list of network resources and their connection, network devices, and their link-state, network device configurations, and their allocated and free resources. The integrated resource management system cannot respond to network failure if it does not know the network topology.

Both systems use OpenStack for their IaaS. They did not mention if they used other solutions to prepare IaaS. An integrated resource management system should be able to use every IaaS solution. Both systems did not mention any consideration to stable hypervisor, which is very important for resource abstraction.

3. Managing IaaS with SDI Management and control system

As mentioned before, current approaches in cloud computing, which use three separate resource management (compute, network and storage) are not capable to respond to the QoS requirements of applications and multimedia services [7]. End-to-end appli-cations and QoS depend on the performance of computing, network, and storage re-sources; hence, these resources should be managed in concert.

Integrated management of heterogeneous resources in the data center provides new management capabilities. Optimized use of cloud infrastructure resources is the most important feature of the integrated resource management system. This system allo-cates resources to the workloads automatically and based on workloads need. In this section, we propose our integrated heterogeneous resource management system accord-ing to our objectives and criteria.

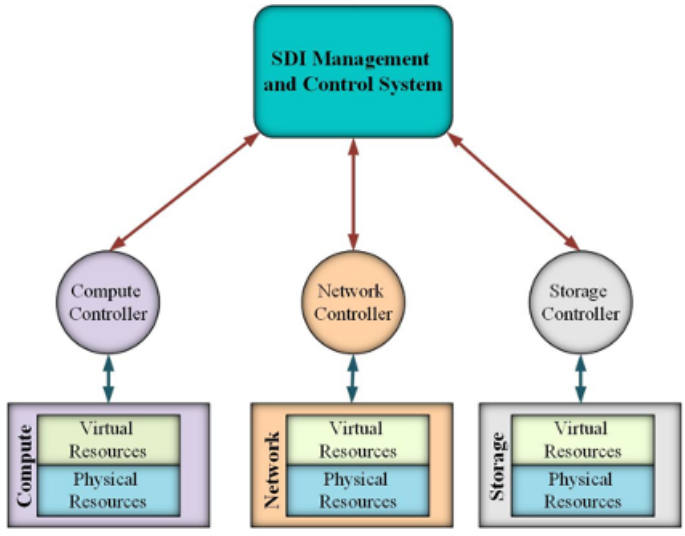

Fig. 3. SDI MCS level 0 architecture

3.1. SDI MCS Architecture

SDI MCS tries to integrate the management of IaaS resources. Figure 3 shows the SDI MCS level 0 architecture. With the knowledge of all the resources in the infrastructure, this system can control and manage resource allocation to workloads. The compute, network, and storage resources are integrated and managed through this system. The resources consist of both physical and virtual resources. SDI MCS receives all the events from compute, network and storage controllers and allocates resources to workloads based on predefined policies and patterns. The processes are distributed among multiple machines. In this system resource, allocation takes au-tomatically and on-demand. If a workload finishes its job, it will free the allocated resources.

SDI MCS is capable of using the main functions of converged heterogeneous resource management. The main functions include: fault tolerance, green network, optimization of resource scheduling, VM migration with network-aware, QoS, workload orchestration and abstraction, flexible detection based on network topology information, and input task management. The Cloud Controller is responsible for computing resources management, VM placement, memory allocation, etc. The network controller takes a network specification and translates it into high-level configuration commands that can be installed on SDN-enabled networking devices. The storage controller also separates the storage control plane from the storage data plane. The automatic re-sponse

of heterogeneous storage resources to on-demand workload requests is one of its ad-vantages. The SDI MCS uses a compute controller for providing computing resources, VM migration, and load balancing. It uses a network controller for controlling, managing, and configuring network resources. It also uses a storage controller to manage storage space and storage allocation to workloads. The SDI MCS performs all the management based on the data it receives from computing, network, and storage controllers.

Figure 4 shows some examples of computing, network, and storage controllers. Nova (OpenStack Nova Homepage, 2017) / VMware vCenter (Center Server Virtualization - Server Management Software: VMware, 2017) prepare to compute resources (the number of free processors and memories) to the SDI MCS, and accord-ingly, the system can reserve resources for incoming workloads. NSX (NSX Homepage, 2017) / Neutron (OpenStack Neutron Homepage, 2017) provides network data for the SDI MCS. They use the OpenFlow protocol to communicate with the system. The network controller pulls all the events from the OpenFlow enabled de-vices and sends them to the SDI MCS. The system can use Swift (OpenStack Swift Homepage, 2017) / Ceph (Ceph Homepage, 2017) for storage controller. These con-trollers are responsible for managing, keeping, and distributing files and objects on heterogeneous storage resources. We developed the OpenDaylight controller and used it at the heart of the SDI MCS.

OpenDaylight has many capabilities, such as OpenStack support, Northbound,

and Southbound API support, high availabilities, scalability, and modularity. We chose OpenDaylight as an appropriate controller at the heart of the SDI MCS because of these capabilities. The Northbound APIs, which are RESTful, can be used on OpenDaylight for IaaS resource management. The compute resource management APIs receive data from the Nova / vCenter and allocate resources to the incoming workloads based on predefined policies and patterns. The SDI MCS distributes the load on servers, network devices, and storage devices based on network topology and data it gains from the controllers. For example, if there is a VM with free computing resources, but physical server bandwidth is occupied, SDI MCS allocates resources to the incoming workload from a server with the lowest load, which has more compute, network and storage resources available.

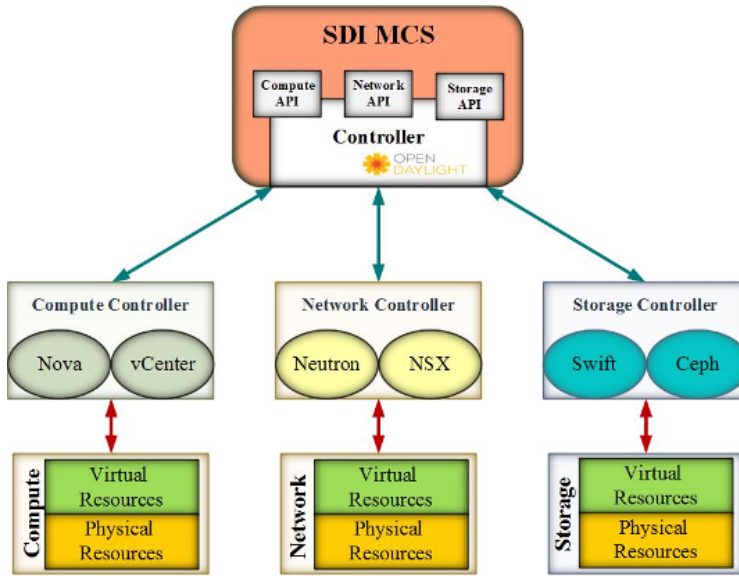

Fig. 4. SDI MCS connection to the IaaS resource controllers

3.2. Design of the System

As mentioned before, we used OpenDaylight at the heart of SDI MCS. The OpenDaylight is modular and supports various protocols. Its language is Java and uses RPC for message communication. Figure 5 shows the SDI MCS reference architecture. At the bottom of this architecture, there are heterogeneous resources, which are combined according to their capabilities and provide a pool of resources for the system. SDI MCS is responsible for mapping the workloads to the resources automatically and optimally based on the results of the workloads and operating conditions of the cloud environment. Several workload schedulers like Control-M (Control-M Workload Automation – BMC, 2017), Jobschedule (JobScheduler | software- und Organisations-Service, 2017), or Rundeck (Rundeck Homepage, 2017) can be used for automation and scheduling the workloads. These tools, manage, au-tomate, and

schedule workloads based on predefined policies and patterns.

Physical compute resources include servers, GPUs, field-programmable gate arrays (FPGAs), and so on. Physical network resources include switches, routers, firewalls, load balancers, and so on. Physical storage resources are local disks on servers, stor-age area networks (SANs), and network-attached storages (NASs). The hypervisor layer abstracts these resources with virtualization and prepares a pool of resources for the system. This system uses virtualization as a first step to optimize the use of re-sources in the cloud infrastructure. In the hypervisor layer, Linux Kernel-based Virtual Machine (KVM), VMware ESXi, Citrix XENServer, and Microsoft HyperV can be used. Upon the hypervisor layer, there are resource controllers. SDI MCS brings to-gether software-defined compute, network, and storage and unify the control and man-agement planes from each software-defined component.

The resource controllers communicate with the system through southbound inter-faces. In OpenDaylight, these southbound interfaces can support various protocols such as OpenFlow, REST, SNMP, and RPC. There are multiple functions in this system, such as computing services functions, billing functions, network services functions, topology manager, and storage services functions. The SDI MCS uses these functions and northbound APIs to allocate resources to the workloads automati-cally and optimally.

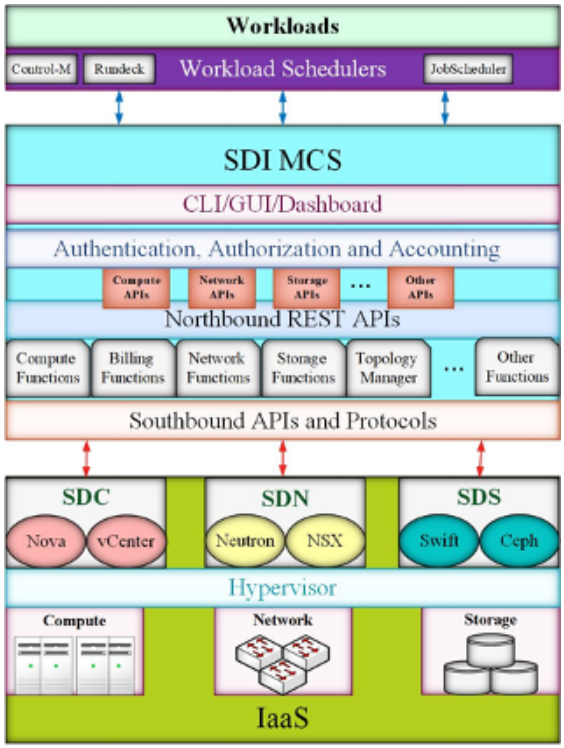

Fig. 5. SDI MCS reference architecture

The SDI MCS monitors the cloud network continuously and reacts appropriately to the network events based on the network topology information it obtains from the topology manager. The topology manager keeps a list of network resources, their relationship, monitoring, and measurement data of each resource and their state. The topology manager is aware of the creation and deletion of the networks, mapping, removing network ports, virtual interface migration from one virtual network to an-other, virtual network control delegation to a person, etc. The topology manager also provides up to date network resources information for SDI MCS, which can manage and control resources based on network topology. It should be mentioned that the received data from the resource controllers are kept in memory instead of the external database to reduce response time for workload resource requests. All data, in order to prevent losing them in the event of a problem, are sent to a SQL database.

The SDI MCS has a pool of heterogeneous resources and allocates them to the workloads through northbound APIs. The proper execution of a workload relates to the appropriate mapping between the workload and infrastructure resources according to the workload-specific objectives and policies. These policies include requirements of high availability, load balancing, automatic VM migration, and scalability to satisfy service-level objectives. The workloads specify the number of resources, and the system allocates the abstracted resources to the workloads automatically and on-demand.

The Authentication, Authorization, and Accounting (AAA) layer authenticates us-ers and applications before they get access to the system based on predefined access and permissions. The system administrators and users use the command line interface (CLI) and graphical user interface (GUI) for management, configuration, and monitoring. The SDI MCS provides a dashboard for users to monitor system status, the incoming workloads, the running tasks, the finished tasks, and the amount of allocated and unallocated resources.

3.3. SDI MCS capabilities

The SDI MCS is a step toward the integration of the IaaS heterogeneous resources. Its capabilities include high performance, scalability, heterogeneous resource integration, resource allocation to the workloads based on the programmable infrastructure, and optimally resource consumption.

The SDI MCS is a unified data center platform that provides automation, flexibil-ity, and efficiency. Compute, storage, networking, security, and availability services are pooled, aggregated, and delivered as software. These services are also managed by intelligent, policy-driven software.

The SDI MCS uses virtualization as a first step to optimize infrastructure re-sources. The traditional view of virtualization focused on the hardware and the bot-tom-up approach. In this approach, compute, network, and storage resource manage-ment is done manually and separately. The SDI MCS is a top-down approach that considers the workloads.

The SDI MCS increases the value of a data center and cloud services. This value which

is achieved by programmable and flexible infrastructures, can be divided into the following categories:

- Decrease response time using the elastic, flexible, and programmable infrastruc-ture.
- Increase stability, fault tolerance, and availability using automatic resource allo-cation to workloads and automatic VM placement.
- Reducing the cost of providing infrastructure and energy.
- Increasing security by isolating allocated resources on the shared infrastructure for multiple tenants.

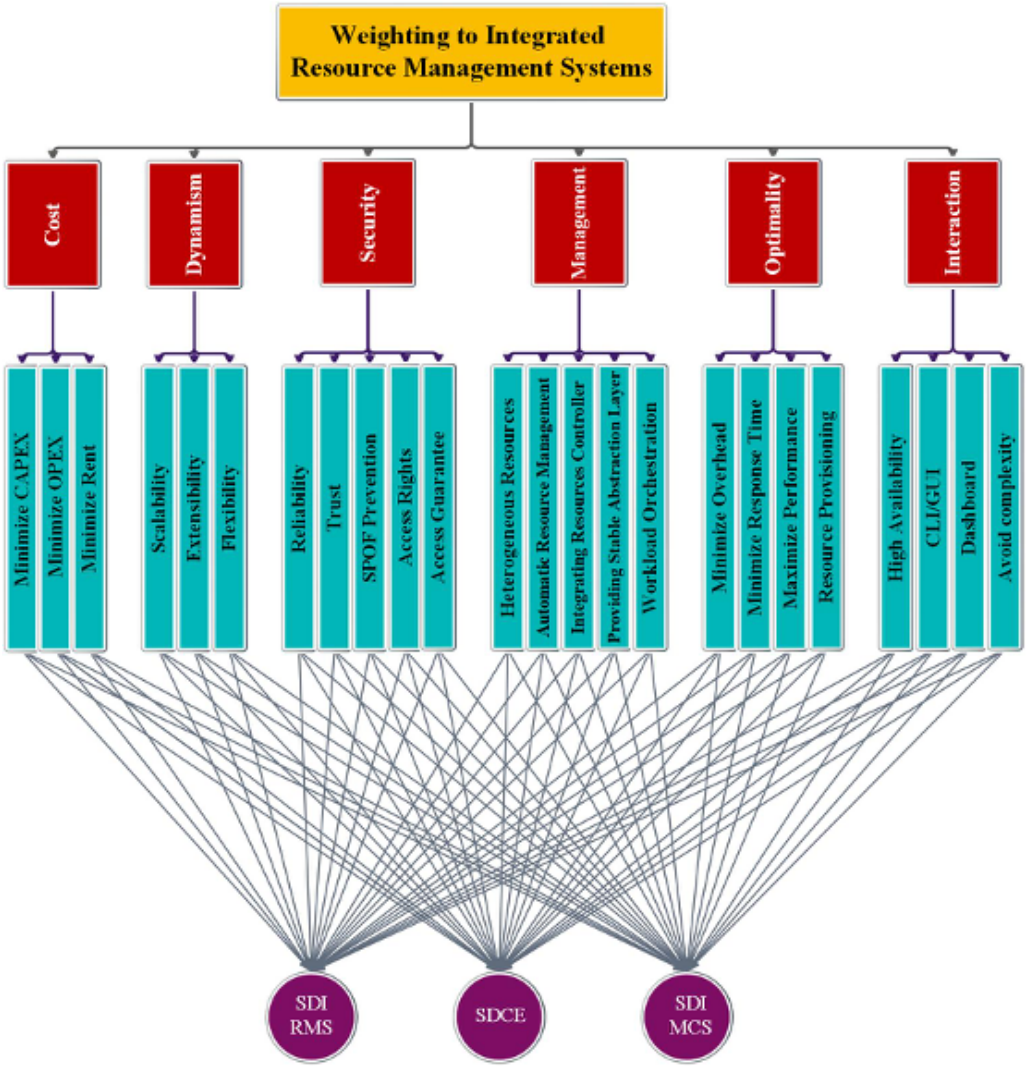

Fig. 6. Integrated resource management system criteria and sub-criteria

4. Evaluation

For evaluating the SDI MCS and comparing it with SDI RMS and SDE, we used the analytic hierarchy process (AHP) (Saaty, T. L., 1990). AHP is a structured tech-nique for organizing and analyzing complex decisions developed by Thomas L. Saaty in the 1970s.

For weighting and prioritizing architectures introduced in this study with AHP, we first need to draw a hierarchical tree that can be seen in Figure 6. At the root of this tree, we place the aim of the analysis, which is "the weighting of integrated resource management systems" here. At level one of this tree is the main criteria: cost, auto-mation, security, management, efficiency, and interactivity. The second level contains the sub-criteria, and options are placed on the third level. In the next step, we made up the pairwise comparison matrices. We made these matrices for the main criteria first and then for all of the sub-criteria of one group over another. In the end, we made the options pairwise matrices for each sub-criteria.

We used the architecture features set, which were mentioned in section II besides the SDI MCS capabilities for weighing the main criteria, sub-criteria, and options.

Table 1. Main criteria pairwise matrix

SDI RMS vs. IBM SDE vs. SDI MCS	Cost	Dyna-mism	Manage-ment	Security	Interac-tion	Optimality
Cost	1	0.25	0.2	0.167	0.333	0.5
Dynamism	4	1	1	0.5	3	1
Management	5	1	1	0.5	4	2
Security	6	2	2	1	5	3
Interaction	3	0.333	0.25	0.2	1	0.333
Optimality	2	1	0.5	0.333	3	1

Table 2. Managing the sub-criteria pairwise matrix

Management	Integrating Resources Controller	Hetero-geneous Resourc-es	Workload Orches-tration	Automatic Resource Manage-ment	Pro-viding Stable Abstrac-tion Layer
Integrating Resources Controller	1	1	2	1	1
Heterogeneous Re-sources	1	1	3	2	2
Workload Orchestra-tion	0.5	0.333	1	0.333	0.25
Automatic Resource Management	1	0.5	3	1	0.5

Providing Stable Abstraction Layer	1	0.5	4	2	1

For example, table 1 shows a pairwise matrix for the main criteria. As shown in this table, security has the highest weight among other criteria. Table 2 shows a pair-wise matrix for managing sub-criteria.

In this research, we used the Expert Choice ver. 11 to calculate the final weights of the main criteria, sub-criteria, and options. To weight main and sub-criteria, we insert the output of all matrices in Expert Choice. The inconsistency rate was 0.04, which shows that paired comparisons were desirable. Figure 7 shows the output of Expert Choice for prioritization of the main criteria based on the objectives.

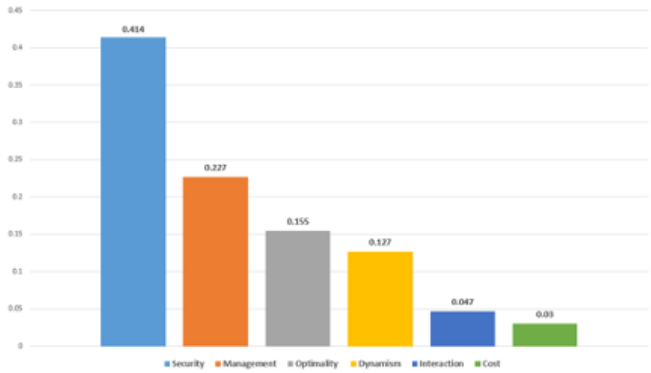

Fig. 7. Main criteria prioritization

The results of eigenvector show that:

- The security criterion has the highest priority, with a normalized weight of 0.414.
- The management criterion has the second priority, with a normalized weight of 0.227.
- Optimality criterion has the third priority with the normalized weight of 0.155.
- Dynamism criterion has the fourth priority with the normalized weight of 0.127.
- The interaction criterion has the fifth priority, with the normalized weight of 0.047.
- The cost criterion has the last priority, with a normalized weight of 0.03.

To prioritize the sub-criteria, we used the same method. Table 3 shows the weights of the main criteria and sub-criteria.

Table 3. The weights of the main criteria and sub-criteria

Main Criteria	Main Criteria Weights	Sub-criteria	Sub-criteria Weights
Cost	0.03	CAPEX Reduction	0.092
		OPEX Reduction	0.423
		Rent Reduction	0.484

Dynamism	0.127	Flexibility	0.433
		Scalability	0.467
		Extensibility	0.1
Security	0.414	Reliability	0.433
		Trust	0.284
		SPOF Prevention	0.143
		Access Rights	0.07
		Access Guarantee	0.07
Management	0.227	Heterogeneous Resource	0.295
		Dynamic Resource Management	0.251
		Integrating Heterogeneous Resource Controller	0.24
		Providing Stable Abstraction Layer	0.091
		Workload Orchestration	0.123
Optimality	0.155	Overhead Reduction	0.148
		Response Time Reduction	0.237
		Performance	0.406
		Resource Provisioning	0.209
Interaction	0.066	Availability	0.556
		Avoid complexity	0.236
		CLI/GUI	0.139
		Dashboard	0.069

To calculate the relative weight for each architecture related to sub-criteria in pairwise matrices, we first summarize column values, and then we normalize matrix elements. Afterward, we calculate the average of the elements in each row. The results are the weights of the options related to the sub-criteria. For example, table 4 shows the weights of options related to scalability sub-criteria.

We did this calculation for every option related to the sub-criteria. We also have the weights of sub-criteria from table 3. The final weights of every architecture can be achieved through the sum of multiplications of the sub-criteria weights to the weights of options. Figure 8 shows the final weights of the resource integrated management systems.

The results achieved from Expert Choice show that SDI MCS has a higher weight than SDI RMS and SDE. SDI RMS has lower weight because it does not consider storage resources and automatic resource allocation to the incoming workloads. SDI

MCS has higher priority than SDE because of its complexity avoidance, the ability to use different controllers, topology management, and OpenStack's independence for providing infrastructure as a service.

Table 4. The weights of options related to the scalability sub-criteria

Scalability	SDI RMS	IBM SDE	SDI MCS	The options weights related to the scalability
SDI RMS	0.285714286	0.333333333	0.272776869	0.297274829
IBM SDE	0.142857143	0.166666667	0.181669394	0.163731068
SDI MCS	0.571428571	0.5	0.545553737	0.538994103

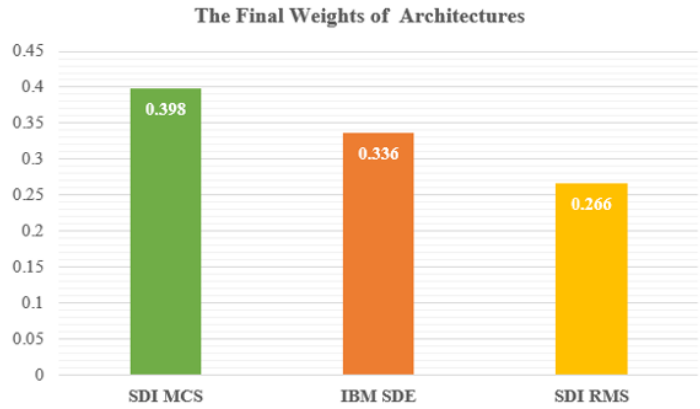

Fig. 8. The final weights of the resource integrated management systems

5. Conclusions

Integrated resource management could prepare dynamic resource allocation to the workloads based on their requirements. In this paper, we first discussed two archi-tectures, SDI RMS and SDE, and then compared them. In the following, we proposed SDI MCS, which was able to integrate the control and management of all the heterogeneous infrastructure resources. This system receives all the events and information from the compute, storage, and network controllers and allocates resources to the workloads based on the predefined policies and patterns. We developed OpenDaylight to be the central part of SDI MCS. The resource controllers communicate with the system through southbound interfaces. There are various functions in this system, and SDI MCS uses these functions and northbound APIs to allocate resources to the workloads automatically and optimally. It also has a pool of hetero-geneous resources and allocates them to the workloads through northbound APIs. A unified

data center platform provides automation, flexibility, and efficiency. Its capa-bilities include high performance, scalability, heterogeneous resource integration, resource allocation to the workloads based on the programmable infrastructure, and optimally resource consumption. For Evaluating the SDI MCS and comparing it with SDI RMS and SDE, we used AHP. The results achieved from Expert Choice show that SDI MCS has a higher weight than SDI RMS and SDE. In future work, we try to optimize the architecture and add more functions to it. We also try to improve the performance and scalability of the system.

References

Arnold, W. C., Arroyo, D. J., Segmuller, W., Spreitzer, M., Steinder, M., & Tantawi, A. N. (2014). Workload orchestration and optimization for software defined environments. *IBM Journal of Research and Development, 58*(2/3), 11:1–11:12.

Bakshi, K. (2013, March). Considerations for software defined networking (SDN): Approaches and use cases. In *2013 IEEE Aerospace Conference* (pp. 1-9). IEEE.

Center Server Virtualization - Server Management Software: VMware, (2017) Retrieved from: http://www.vmware.com/products/vcenter-server.html.

Ceph Homepage (2017) Retrieved from: http://ceph.com/.

Control-M Workload Automation - BMC (2017) Retrieved from: http://www.bmc.com/it-solutions/control-m.html.

Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., .. & Stoica, I. (2009). Above the clouds: A berkeley view of cloud computing. Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, 28(13), 2009.

Javan, M. S., & Akbari, M. K. (2011, November). Cloud Computing Issues and Challenges for Ultimate Interoperability. In *1st CSUT Conference on Computer", Communication and Information Technology, University of Tabriz*.

JobScheduler | software- und Organisations-Service (2017) Retrieved from: http://www.sos-berlin.com/jobscheduler.

Lara, A., Kolasani, A., & Ramamurthy, B. (2013). Network innovation using openflow: A survey. *IEEE communications surveys & tutorials, 16*(1), 493-512.

Li, C. S., Brech, B. L., Crowder, S., Dias, D. M., Franke, H., Hogstrom, M., et al. (2014). Software defined environments: An introduction. *IBM Journal of Research and Development, 58*(2/3), 1-1.

Lin, T., Kang, J. M., Bannazadeh, H., & Leon-Garcia, A. (2014, May). Enabling SDN applications on software-defined infrastructure. In *2014 IEEE Network Operations and Management Symposium (NOMS)* (pp. 1-7). IEEE.

NSX Homepage (2017) Retrieved from: http://www.vmware.com/products/nsx.

OpenStack Neutron Homepage, (2017) Retrieved from: https://wiki.openstack.org/wiki/Neutron.

OpenStack Nova Homepage, (2017) Retrieved from: http://docs.openstack.org/developer/nova/.

OpenStack Swift Homepage, (2017) Retrieved from: https://wiki.openstack.org/wiki/Swift.

Quintero, D., Genovese, W. M., Kim, K., Li, M. J. M., Martins, F., Nainwal, A., et al. (2015). IBM software defined environment. IBM Redbooks.

Rundeck Homepage, (2017) Retrieved from: http://rundeck.org/.

Saaty, T. L. (1990). Decision making for leaders: the analytic hierarchy process for decisions in a complex world. RWS publications.

Singh, A., Korupolu, M., & Mohapatra, D. (2008, November). Server-storage virtualization: integration and load balancing in data centers. In *SC'08: Proceedings of the 2008 ACM/IEEE conference on Supercomputing* (pp. 1-12). IEEE.

ViPR Controller Software-defined Storage | EMC, (2017) http://www.emc.com/vipr.

Submitted 15.03.2020
Accepted 17.05.2020