MC2E: META-CLOUD COMPUTING ENVIRONMENT FOR HPC

V. Antonenko [1, a], I. Petrov [1, b], R. Smeliansky [1, c], Z. Huang [2, d], M. Chen [3, e], D. Cao [4, f], X. Chen [4, g]

[1] *Lomonosov Moscow State University, 1 Leninskiye Gory, Moscow, 119991, Russia*

[2] *Tsinghua University, 30 Shuangqing Rd, Haidian, Beijing, 100084, China*

[3] *Huazhong University of Science & Technology, 1037 Luoyu Rd, Wuhan, 430074, China*

[4] *Peking University, 5 Yiheyuan Rd, Haidian, 100871, China*

E-mail: [a] anvial@lvk.cs.msu.su, [b] ipetrov@cs.msu.ru, [c] smel@cs.msu.ru, [d] huangzc@tsinghua.edu.cn, [e] minchen2012@hust.edu.cn, [f] caodg@pku.edu.cn, [g] cherry@pku.edu.cn

Recently in many scientific disciplines, e.g. physics, chemistry, biology and multidisciplinary research have shifted to the computational modeling. The main instrument for such numerical experiments has been supercomputing. However the number of supercomputers and their performance grows significantly slower than the growth of users' demands. As a result, users may wait for weeks until their job will be done. At the same time the computational power of cloud computing grow up considerably and represent today a plenty available resources for numerical experiments for many applications. There are several problems related to cloud and supercomputer integration. First, is how to make a decision where to send a computational task: to a supercomputer or to cloud. Second, various platforms may have significantly different APIs, and it's often labor-expensively for researchers to move from one platform to another since it would require large code modification. In this research we present MC2E – an environment for academic multidisciplinary research. MC2E aggregates heterogeneous resources such as private/public clouds, HPC clusters and supercomputers under a unified easy-to-use interface. This environment will allow to schedule parallel applications between clouds and supercomputers based on their performance and resource usage. MC2E will also provide a Channel-on-Demand service to connect clouds and supercomputers with channels that are used to send data for parallel applications.

Keywords: High Performance Computing, Supercomputer, Cloud, Data-Center

Vitaly Antonenko, Ivan Petrov, Ruslan Smeliansky, Zhen Chun Huang,
Min Chen, Donggang Cao, Xiangqun Chen

1. Introduction

Today's research in various fields such as physics, chemistry and biology have shown large demands in computational resources due to the complexity of tasks performed. Such resources are often provided as supercomputers and clusters for High Performance Computing (HPC). The general trend today is the use of supercomputers or HPC installations. However, a trend analysis at TOP500.org [6] suggests that the number of applications is growing faster than the number of supercomputers and HPC installations. At the same time, we can see the rapid growth in the popularity of cloud computing, the usage of data centers (DC) networks (DCN) to increase the power of cloud computing platforms. A good example is EGI Association [7]. These two kind computational platforms have a different computational capability but they also have big differences in their load. Most applications will run faster on a supercomputer than on a server cluster in a DC. However, it may turn out that the total delay of the application in the queue plus the execution time may turn out to be more than a longer execution on the server cluster, but with a shorter waiting time in the queue.

These considerations lead us to the idea of the integration of these two pretty different environments – HPC supercomputers and DC Clouds. These environments vary in many ways: differences in the level of resource management in the computational environment in use, by the virtualization technique, by the composition of the parameters and the specification form of the request to execute the application (task), by scheduling and resource allocation policy. On-demand clouds could help solve this problem by offering virtualized resources customized for specific purposes. Cloud platform offers more flexibility and convenience for researchers. But in any case the HPC and Cloud platforms heterogeneity makes it hard to switch automatically between them if some platform becomes highly loaded or inaccessible. For example, because of different interfaces (APIs) for task submission. So in order to change the target platform, researchers need to spend time and resources adjusting their software for the new API.

Another problem on the way to the integration of the HPC-Supercomputer (HPC-S) and the HPC cloud server cluster (HPC-C) is the automation of the recognition in the queue of tasks to HPC-S of those that can be solved in HPC-C in the current resource amount/configuration and, therefore, transferred to the HPC-C queue as a request for services.

For the problem above we need to justify the basis of the hypothesis that the virtualization technique, which is actively used in the HPC-C environment and involves the sharing of physical resources by several tenants in HPC-C environment, will provide the expected effect for HPC tasks.

One more problem is the ability of HPC-C environment aggregates the resources of DCN. At this point the key problem is feasibility the Channel-on-Demand (CoD) service problem. This service should develop/allocate, on request, the channels between two or more DC with the appropriate QoS parameters and throughput for transmitting a given amount of data at a limited time without creating a physical channel between them in advance, i.e. through aggregation of existing communication resources.

In this paper we present the MC2E project intended to find the solutions for the listed above and develop the environment for multidisciplinary academic research that aggregates heterogeneous resources such as private/public clouds, HPC clusters and supercomputers under a unified easy-to-use interface. Comparing with “traditional” resource orchestration in DC, that use open source tools like OpenStack [8] or commercial provided by VMware [9], MC2E offers a number of new features/opportunities and advantages:

- an aggregated resource control (resources of the multiple platforms instead of a single one in a local DC or HPC cluster);
- flexible capabilities to define virtual environments, more types of resources and services;
- high quality of resource scheduling and utilization;
- relieves users from tedious system administration tasks;

- a unified way to describe and support virtualized services (NFV) life cycle in DC (or HPC cluster), to apply existing user's software to performing experiments on MC2E infrastructure.

MC2E enlarges the concepts of PaaS and IaaS to scientific applications area. We believe that it could be of great help to research teams that work jointly and need a shared virtual collaboration environment with resources from different geographically distributed platforms. MC2E project is a joint effort of the following parties:

- Lomonosov Moscow State University (Russia);
- Tsinghua University (China);
- Huazhong University of Science & Technology (China);
- Peking University (China).

The paper structure is the following. Section 2 presents multidisciplinary research problems description. Section 3 describes proposed solution. Section 4 contains a detailed description of the MC2E components. Section 5 presents an acknowledgement. Section 6 depicts the expected results of the MC2E project.

2. Problem Description

Modern interdisciplinary research is often performed using unique scientific instruments by multiple research teams in collaboration. Such collaboration requires an informational, computational and communication infrastructure specifically tuned for each project. Efforts to create such infrastructure in a traditional way (a local DC or a HPC-C with domain-specific communication software) cause a number of problems:

1. It requires significant financial and material investments, since each new experiment needs specific software adjusted by highly qualified IT-specialists;
   The problem becomes more complicated if such experiments are performed by independent research teams, since such teams often have different internal business processes, specialize in different subject areas, have their own hardware and software preferences and are may be located far from each other;
2. At the initial stage of a project the requirements to the project infrastructure are known only with some approximation and often overestimated. Infrastructure developers often consider the worst cases when estimate resources for the project. This could lead to resource under-utilization and thus waste the efficiency of investments;
3. A lot of difficulties also arise when scientific data is distributed and used by different teams simultaneously;
4. Data that is needed for one team could be acquired by another. And without a specialized system that manages infrastructure such cases are hard to solve;
5. The groups of researchers from different projects may already have tools, software for processing, collecting and storing data. Creating or mastering new ones for a project is usually unacceptable. Therefore, it is necessary to provide the possibility to bring into the environment already existing developments, using for this technology, for example, network function virtualization (NFV).

3. Proposed Solution

In order to solve problems described above we propose Meta-Cloud Computing Environment (MC2E). This environment is based on the following principles:

- The infrastructure is created as a *federation* of local computing units called *federates*. Such federation controls all resources (CPU, memory, network, software) provided by underlying federates;
- All physical resources are virtualized;
- Resources of a single federate can be shared between different projects simultaneously;
- Resources have a high level of abstraction. Using such resources should not require a system administrator qualification;
- Experiments' results could be saved. Such saved results could be used by other research teams to reproduce or continue the experiment;
- The federation provides data processing as a virtual service.

A federate could be an HPC cluster, data-center, supercomputer, scientific instrument or a tenant in a cloud. And each federate has its own policy that defines how such federate delegates its resources to the users of the federation. Infrastructures that are built as federations of heterogeneous computational resources are already used in many existing projects. Several such projects are designed to perform experiments in computer networking. For example, the GENI project (Global Environment for Network Innovations) [1] that was initiated by the US National Scientific Foundation (NSF) is a virtual laboratory aimed to provide an environment for networking experiments on an international scale. Today more than 200 US universities contribute to the GENI project. Similar but less known projects are Ophelia [11] (supported by the UN) and Fed4Fire [12] (supported by 17 companies from 8 countries). Other projects provide environments for performing different computational experiments regardless of their domain. Such projects are Open Science Data Cloud [13], ORCA [14] and GEANT [15]. However, these projects have several significant limitations:

- The lack of protocols for interaction between heterogeneous federates (for example, HPC-S and HPC-C);
- The lack of a specialized language for describing services required to perform experiments and bring already existed tools into the new environment;
- Resource planning doesn't take into account possible services' scaling;
- The lack of billing system that allow a decentralized resource accounting and mutual settlements between project participants.

In this project we propose to develop a virtual infrastructure for multidisciplinary research. The proposed infrastructure will be based on Software-Defined Networking (SDN) [16] and Network Function Virtualization (NFV) [17]. This approach will increase the resource abstraction, enable coordinated resource optimization and automatize infrastructure management.

We plan to implement the proposed environment based on stretching the concept of network and HPC service virtualization. Instead of providing individual resources, users receive complete virtual infrastructures (computing power, communication channels and storage systems) with guaranteed performance a QoS based on the service level agreement (SLA). The proposed federation based environment will have the following advantages:

1. Easy scaling;
2. Merging infrastructures from different research teams and adjust access policies;
3. Automated resource planning for fulfilling user requests based on access policies and SLA;
4. Extensive application description environment, that allows to abstract away low level system details;
5. A decentralized resource accounting system for settlements between project participants;
6. Wider possibilities for experiments' tracing and monitoring compared to a general data-center;

7. Increased efficiency of network virtualization with SDN, that allows to adjust virtualized network channels for each particular experiment;
8. Common specification language that is necessary for transferring research software into MC2E;
9. The environment could also be used for education purposes, since it will allow students to study new methods and technologies used for scientific experiments.

4. MC2E Architecture

This section contains a detailed description of main MC2E architecture components:

1. *Meta-Cloud (*the most important) – performs application scheduling between federates;
2. *Interface* – provides a unified API for users to submit parallel applications and for federate maintainers to include their resources into federation;
3. *Networking* – regulates network resource distribution and provides Channel-on-Demand [10];
4. *Monitoring* – performs resource monitoring and clearance for all federates included in MC2E;
5. *Quality of Service, Administration Control and Management* – enforces resource usage policy, provides QoS based on user requirements and guarantees resource reliability.

The key principle of MC2E is to distribute parallel applications between HPC-S, HPC-C and data-centers. This application distribution is aimed to even the load on different federates and to minimize queue waiting time for users. Applications are distributed based on their performance on different platforms, their network usage and data size. General MC2E workflow looks as follows:

1. User uses a unified MC2E interface to send an application (with data) to the front-end server;
2. The front-end server invokes MC2E scheduler and monitoring to choose federate for application execution;
3. MC2E retrieves queue sizes and predicts application performance and data transmission time for all available federates;
4. Based on the prediction MC2E chooses the federate that will minimize total execution time (data transmission time + queue waiting time + execution time);
5. MC2E network creates a channel to the destination federate and sends application data;
6. Federate executes the application and returns results to the user;
7. In the case of a federate failure, MC2E QoS migrates the application to another federate.

4.1. Meta-Cloud

Meta-Cloud consists of three components which are described below in separated sections.

4.1.1. MC2E MANO system

This Management & Orchestration System is intended to support systematically complete life-cycle of a service in MC2E accordingly with ETSI standard recommendations. This system will also responsible for service instance scaling and service instant healing support. The system will provide a flexible way to integrate new services into the managed infrastructure.

4.1.2. MC2E Resource Scheduling

The special optimization techniques and algorithms should support scheduling in heterogeneous environment (HPC or DCs), providing consistent scheduling of different types of resources (network, storage and compute). These algorithms should accept the set of limitation of resource usage, that we call Service Level Agreement (SLA) and the set of deploying policies like VM-VM, VM-PM affinity/anti-affinity. These algorithms should take into consideration the management policy of the specific service. For example, compute node horizontal scaling policy in MPI task of HPC cluster, or scaling and healing policies in network DC services (e.g. Firewall, NAT, Load Balancing).

4.1.3. Enhanced MC2E Service Orchestrator

The enhanced Orchestrator should distinguish which MC2E service should run on DC infrastructure and which on HPC one. The decision will be based on task requirements and the current state of MC2E infrastructure. The goal is to prevent computing task on high-price HPC unit whether it can be easily computed in DC environment. For example, MPI or Spark task is better to deploy in HPC infrastructure and network function NAT in DC.

4.2. Interface

4.2.1. MC2E Virtual Resource Description API

The subsystem that should support the unified specification of virtual environments in DC (NATs, LBs, Web-servers etc.) and in HPC (like MPI, Spark/Hadoop etc.) infrastructure. The descriptions will also include data that will help to manage (scale, heal and configure) the virtual resource. These specifications should be available in all forms listed below: GUI, CLI and REST APIs.

4.2.2. MC2E to Other Cloud Initiatives Gateway

There are a number of other cloud initiatives in the world; our proposal is to provide a gateway to interconnect with them, for example, NSF Cloud Initiative, Amazon Web Services, and Rackspace among others. Our intent is to investigate API compatibility and propose a gateway to translate signaling and provisioning among public and scientific cloud services and MC2E. In such a way that computation or storage resources can be moved to other clouds smoothly. Also, Party 1 will provide the connectivity the infrastructures of Russian Space Research Institute and Joint International Nuclear Research Institute to MC2E.

4.2.3. MC2E Virtual Cloud Workspace

Users need a convenient tool to help build their custom virtual cloud. MC2E Virtual Cloud Workspace is a WEB-based system that users can use to manage resources and run tasks. A workspace is the portal of a user's own virtual cloud supported by the resources allocated to the user. With a browser supporting HTML5, users can do jobs like online coding, debugging, testing, running program and analyzing results in their workspaces. A virtual cluster manager will manage the supported resources (from DCs and HPC units) as a virtual cluster. This virtual cluster should be elastic, fault tolerant and provided as a service to users.

4.2.4. HPC Gateway

To run HPC jobs in some HPC unit through MC2E, we need to develop the corresponding HPC gateway first. This gateway should provide HPC resources as services to MC2E resource manager, and acts as a proxy of the HPC unit to run HPC jobs for MC2E users. This gateway may also provide services like logging, accounting, billing, etc.

4.3. Networking

4.3.1. Classifying Network Services for MC2E and Inter-Communication

In practice, some services are not intended to be implemented as a virtual machine in the cloud and to be placed on MC2E switch or some other network equipment. In order to be able to find the proper way to place the Network service, we need to produce the classification of the Network services based on the service infrastructure requirements and service life-cycle limitations. Some of them could be implemented on MC2E SDN controller, or as a virtual appliance in a DC.

4.3.2. MC2E Component Stitching

As long as an MC2E federation typically combines resources from different locations, there is a demand in a framework that can provision appropriate communication paths to interconnect them. In order to resolve this issue, we propose to integrate into MC2E a subsystem for inter-domain traffic engineering (TE) (similar to RouteFlow used by Google B4) based on either MPLS or MP-BGP connected to several IXPs.

4.3.3. Segmenting MC2E intra-federation transport connections

In MC2E end-to-end connections would typically run through DC, HPC and WAN networks, which have different link quality (i.e. bandwidth, delay, loss, and jitter), different equipment facilities (i.e. packet scheduling and AQM disciplines) and different balancing techniques (i.e. ECMP and multipath routing). We suggest splitting these connections into a set of shorter connections with the help of enhancing proxies allocated at the borders of these networks. This approach makes it possible to select the most appropriate congestion control algorithm within each network segment, and, thereby, increase overall speed of end-to-end TCP connections.

4.3.4. Cognitive SDN based MC2E Orchestration

In order to improve the resource and energy efficiency of MC2E, we propose the architecture of cognitive SDN (Software-Defined Network). Based on the advanced technologies of cognitive engine with learning and decision capabilities as well as the interaction with SDN controller, the intelligence offered by cognitive SDN is used to achieve MC2E orchestration. For MC2E management, cognitive SDN located in HPC unit links with multiple DCs, enabling resource-efficient content delivery and large scale virtual machine migrations.

4.3.5. MC2E HPC and DC Communication Normalization

Inside a DC within MC2E, there could be elements that are very specialized for High Performance Computing, like machines connected to InfiniBand switches for example, while other elements are just regular commodity hardware consolidating virtual machines. Thus, there is a need to normalize the communication interfaces and protocols of these components. Thus, our idea is to investigate virtualized version of HPC interfaces that will appear directly in the virtual machines, and further accelerate transport (based on DPDK) of HPC specific protocols from Ethernet to HPC, and further transformation of this communication from HPC to DC, back and forth.

4.4. Monitoring

4.4.1. MC2E Monitoring System

To produce human-readable information about every physical and virtual entity in MC2E infrastructure the flexible monitoring system should be developed. This system should work in real-time environment and have APIs to connect with well-known infrastructure monitoring systems such as Zabbix [19], or NAGIOS [20].

4.4.2. MC2E Clearance System

For the purpose of the Federation resource usage, we need to be able to count of the amount of each resource type each federate has used, providing clearing, billing, and monitor information.

4.5. Quality of Service, Administration Control and Management

4.5.1. MC2E Federation and Federate Resource Usage Policy

The resources (compute, storage and network) of each federate should be divided into two pools. First are local resources of the federate. Second are the resources that the federate delegates to the Federation. To organize the collaboration of federates in the Federation there should be a policy – a specific set of rules that clarify and describe the resource announce/sharing/unsharing processes.

4.5.2. User-oriented QoS Provisioning for resource-consuming Scientific Computing

A MC2E carries various kinds of requests with different importance or priority levels from many individual users. The QoS provisioning should be differentiated among different users. Even for the same user, the QoS requirements can change dynamically over time. From the multi-client QoS support point of view, the traditional cloud system is insensitive to various QoS requirements for a large number of clients coming from different countries, especially for scientific computing tasks with inherent features of workload variations, process control, resource requirements, environment configurations, life-cycle management, reliability maintenance, etc.

4.5.3. Survivability/reliability of MC2E

In case of a cloud failure, MC2E should guarantee uninterrupted communications to offer almost uninterrupted services. Thus, it is very crucial to design finely tuned redundancy to achieve the desired reliability and stability with the lowest resource waste.

5. Conclusion

In this research we present MC2E – an environment for academic multidisciplinary research. MC2E aggregates heterogeneous resources such as private/public clouds, HPC clusters and supercomputers under a unified easy-to-use interface. MC2E is built as a federation of local computing units called federates. Each federate can be represented as an HPC cluster, a data-center, a supercomputer, a scientific instrument or a tenant in the cloud. The advantages of MC2E include:

- High level of resource control and flexible capabilities to define virtual environments;
- High quality of resource scheduling and utilization;
- It relieves a user from tedious system administration tasks and it also specifies a unified way to describe a data center (or an HPC cluster) service livecycle.

MC2E can be applied in different areas, such as educational activities of research institutes and universities, interdisciplinary research, international research collaboration, increasing resource utilization in data-centers, popularizing supercomputer usage in different research areas, shared data usage by multiple organizations.

Acknowledgement

This work is supported by Russian Ministry of Science and Higher Education, grant #05.613.21.0088, unique ID RFMEFI61318X0088 and the National Key R&D Program of China (2017YFE0123600).

References

[1] Hwang, T. (2017, March). NSF GENI cloud enabled architecture for distributed scientific computing. In 2017 IEEE Aerospace Conference (pp. 1-8). IEEE.

[2] Kogias, Dimitrios G., Michael G. Xevgenis, and Charalampos Z. Patrikakis. "Cloud federation and the evolution of cloud computing." Computer 49.11 (2016): 96-99.

[3] Comi, Antonello, et al. "A partnership-based approach to improve QoS on federated computing infrastructures." Information Sciences 367 (2016): 246-258.

[4] Assis, Marcio RM, and Luiz Fernando Bittencourt. "A survey on cloud federation architectures: Identifying functional and non-functional properties." Journal of Network and Computer Applications 72 (2016): 51-71.

[5] Antonenko, V., et al. "Towards SDI-bases Infrastructure for supporting science in Russia." 2014 International Science and Technology Conference (Modern Networking Technologies)(MoNeTeC). IEEE, 2014.

[6] Meuer, H., et al. "The Top500 project." URL: http://www.top500.org/ (2019).

[7] Kranzlmüller, D., de Lucas, J. M., & Öster, P. (2010). The european grid initiative (EGI). In Remote instrumentation and virtual laboratories (pp. 61-66). Springer, Boston, MA.

[8] Sefraoui, O., Aissaoui, M., & Eleuldj, M. (2012). OpenStack: toward an open-source solution for cloud computing. International Journal of Computer Applications, 55(3), 38-42.

[9] VMWare products. https://www.vmware.com/products.html (2019).

[10] Mahimkar, A., Chiu, A., Doverspike, R., Feuer, M. D., Magill, P., Mavrogiorgis, E., ... & Yates, J. (2011, November). Bandwidth on demand for inter-data center communication. In Proceedings of the 10th ACM Workshop on Hot Topics in Networks (p. 24). ACM.

[11] Dewar, R. G., MacKinnon, L. M., Pooley, R. J., Smith, A. D., Smith, M. J., & Wilcox, P. A. (2002). The OPHELIA Project: Supporting Software Development in a Distributed Environment. In ICWI (pp. 568-571).

[12] Fed4Fire project: https://www.fed4fire.eu/the-project/ (2019)

[13] Grossman, R. L., Gu, Y., Mambretti, J., Sabala, M., Szalay, A., & White, K. (2010, June). An overview of the open science data cloud. In Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing (pp. 377-384). ACM.

[14] Bal, H. E., Bhoedjang, R., Hofman, R., Jacobs, C., Langendoen, K., Rühl, T., & Kaashoek, M. F. (1998). Performance evaluation of the Orca shared-object system. ACM Transactions on Computer Systems (TOCS), 16(1), 1-40.

[15] Brun, R., Urban, L., Carminati, F., Giani, S., Maire, M., McPherson, A., ... & Patrick, G. (1993). GEANT: detector description and simulation tool (No. CERN-W-5013). CERN.

[16] Kreutz, D., Ramos, F., Verissimo, P., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2014). Software-defined networking: A comprehensive survey. arXiv preprint arXiv:1406.0440.

[17] Hawilo, H., Shami, A., Mirahmadi, M., & Asal, R. (2014). NFV: State of the art, challenges and implementation in next generation mobile networks (vEPC). arXiv preprint arXiv:1409.4149.

[18] MC2E Project. http://www.fcpir.ru/participation_in_program/contracts/05.613.21.0088/ (2019).

[19] Zabbix, S. I. A. (2014). Zabbix. The Enterprise-class Monitoring Solution for Everyone.

Barth, W. (2008). Nagios: System and network monitoring. No Starch Press.