2013 IEEE International Conference on Cloud Computing Technology and Science

ICOMF: Towards a multi-cloud ecosystem for dynamic resource composition and scaling

Ana-Maria Oprescu*, Alexandru-Florian Antonescu†‡, Yuri Demchenko*, Cees de Laat*
*SNE, Universiteit van Amsterdam, The Netherlands
†SAP (Switzerland) Inc., Althardstrasse 80, 8105 Regensdorf, Switzerland
‡University of Bern, Communication and Distributed Systems (CDS), Neubrückstrasse 10, 3012 Bern, Switzerland

{a.m.oprescu,y.demchenko,delaat}@uva.nl,alexandru-florian.antonescu@sap.com

***Abstract*—Modern cloud-based applications and infrastructures may include resources and services (components) from multiple cloud providers, are heterogeneous by nature and require adjustment, composition and integration. The specific application requirements can be met with difficulty by the current static predefined cloud integration architectures and models. In this paper, we propose the Intercloud Operations and Management Framework (ICOMF) as part of the more general Intercloud Architecture Framework (ICAF) that provides a basis for building and operating a dynamically manageable multi-provider cloud ecosystem. The proposed ICOMF enables dynamic resource composition and decomposition, with a main focus on translating business models and objectives to cloud services ensembles. Our model is user-centric and focuses on the specific application execution requirements, by leveraging incubating virtualization techniques. From a cloud provider perspective, the ecosystem provides more insight into how to best customize the offerings of virtualized resources.**

I. Introduction

In computational science, parameter sweep or bag-of-tasks applications are as dominant as computationally demanding. An ideal match for these demands can be seen in commercial cloud offerings, like Amazon's EC2, where computers can be allocated ("rented") for given time intervals. The various commercial offerings differ not only in price, but also in the types of machines that can be allocated. While all machine types are described in terms of CPU clock frequency and memory size, it is not clear at all which machine type can execute a given user application faster than others, let alone predicting which machine type can yield the best price-performance ratio. The problem of allocating the right number of machines, of the right type, for the right time frame, strongly depends on the application program, and is left to the user.

With the recent Google Compute Engine [1] and a plethora of smaller scale players [2], [3], the cloud computing market is no longer a synonym of Amazon EC2. While this current state of the market encourages further technical advancements and ensures progress, there is a new problem: customers feel lost. With so many cloud providers offering their services at different layers (IaaS, PaaS, SaaS, DaaS, XaaS), the user (customer) finds it very difficult to manually select the best option for the application at hand. When such a task has to be repeated under different constraints, the decision should consider one very important factor: the relative efficiency of the chosen cloud provider(s). Here, two types of issues occur: first, the standardization of cloud benchmarks is still in its infancy, and secondly, the metrics needed to assess the relative efficiency are application dependent.

Ideally, the customer should be able to acquire the needed application execution components from different cloud providers. This has brought forward the need for a multi-cloud architecture. A number of recent research efforts [5], [6] have been dedicated to bridging the gap between single cloud provider design to multi-cloud providers systems. In our previous work [7], [8], we have defined a generic reference architecture for such multi-cloud systems, the InterCloud Architecture Frameworks (ICAF). The focus of ICAF has been mostly on the interactions between different cloud providers.

In this paper, we introduce a dynamic resource composition and scaling multi-cloud ecosystem, ICOMF, designed to support the operation and management layer of the Intercloud architecture framework (ICAF). Here, we focus on the interactions between the customers and the cloud providers from the user perspective. To that end, we explored typical use-cases, namely bag-of-tasks execution and scientific web services, when using multiple cloud providers. Our findings helped us shape the guidelines that should govern such an ecosystem.

The paper is organized as follows: in Section II we summarize the existing multi-cloud architecture designs and identify the mature technologies that would provide the required support for our ICOMF design. Section III provides background information about the ICAF design and the BaTS scheduler, as well as revisiting the Pareto optimality concept. Section IV describes how a popular application type, i.e. task farming can be deployed in a multi-cloud environment. Section V describes our proposal, while Section VII presents our conclusions.

II. Related Work

We relate our work to several perspectives: the cloud market and its business models, the models describing the participating parties and the multi-level service level agreement and (re-)negotiation.

Intercloud architecture and multi-cloud issues have been addressed by a number of research efforts [9], [10], [11]. However, they each focus on different aspects of Intercloud services provisioning and integration.

Market-oriented computing has been recently introduced as a cloud-related research topic [12], based on previous work [6].

978-0-7695-5095-4/13 $31.00 © 2013 IEEE
DOI 10.1109/CloudCom.2013.14

Our vision differs in that we introduce an advisory entity that would benefit both the customer and the provider of such a market. We also aim to design an ecosystem where cloud offerings have a larger degree of flexibility and customization.

The insufficiency of current static cloud offerings and virtualization packages has been identified by Wright et al. [13]. In the same project, a provider-agnostic (virtual) resource discovery API is proposed; the user, however, is assumed to already know the types of resources that best fit her application. Moreover, the performance/storage/network models used to help estimate the costs are predefined. Another recent initiative [14] urges the cloud providers to transition towards a Resource-as-a-Service (RaaS) paradigm. Our work complements this initiative in that we focus on the (RaaS) users' perspective and subsequently put forward the architecture of the recommendation systems such users would employ.

A recent initiative from Cloudstack [15] undertakes dynamic (at runtime) scaling of virtual machines. Such initiatives will provide the required technological support for a dynamic cloud ecosystem and a shift towards a RaaS paradigm.

To describe the business model of our ecosystem, we adopt the Resource-Ownership-Role-Action (RORA) model [16] as proposed and implemented in the Geysers project [17].

The need for dynamic SLA-rewriting has been motivated by previous research efforts [6]. We adopt as a reference architecture for service level agreement management the work of Happe et. al. [18]. The multi-level SLA fits best with our proposed ecosystem, given the vertically (IaaS, PaaS, SaaS, DaaS components) dynamic nature of the cloud offerings.

III. Background

We briefly describe two existing systems based on which we developed the design of ICOMF: the Intercloud Architecture Framework and BaTS, the multi-cloud budget-aware scheduler. We also briefly revisit the Pareto optimality concept.

A. The Intercloud Architecture Framework

We begin by introducing our previous work on a reference architecture for an intercloud system, the Intercloud Architecture Framework (ICAF). ICAF [7] addresses the interoperability and integration issues in the current and emerging heterogeneous multi-domain and multi-provider clouds that could host modern and future critical enterprise and e-Science infrastructures and applications, including integration and interoperability with legacy campus/enterprise infrastructure. To that end, ICAF proposes a multi-layer Cloud Services Model that combines commonly adopted cloud service models, such as IaaS, PaaS, SaaS, in one multi-layer model with corresponding inter-layer interfaces. The ICAF definition is using a general case of provisioning the cloud based infrastructure to support the enterprise or scientific workflow and operations related to the processes monitoring and data processing. Cloud technologies simplify building such infrastructure and provisioning it on-demand.

The ICAF includes the following components that separate all functions related to the cloud services design, control, management and operations into orthogonal groups:

- Intercloud Control and Management Plane (ICCMP), detailed in Figure 1(right), is responsible for Intercloud applications/infrastructure control and management (e.g. inter-applications signaling, synchronization and session management, configuration, monitoring, runtime infrastructure optimization including VM migration, resources scaling, and jobs/objects routing);
- Intercloud Federation Framework (ICFF), detailed in Figure 1(left), is responsible for enabling federation of independent clouds and/or infrastructure components; this should support federation at the level of services, business applications, semantics, and namespaces, assuming necessary gateway or federation services.
- Intercloud Operation and Management Framework (ICOMF), detailed in Section V, includes functionalities to support multi-provider infrastructure operation (e.g. including business workflows, SLA management and accounting). ICOMF defines the the actors as well as their roles and interactions in sense of resources ownership, operation, and management. ICOMF requires support from, and interacts with both ICCMP and ICFF.
- Intercloud Security Framework (ICSF) provides a basis for secure operation of all components of the Intercloud infrastructure; ICSF should integrate the security services of different CSM layers and all participating cloud service providers.

B. The BaTS scheduler

Here we describe our previous work on heterogeneous virtual resources scheduling. BaTS [19] schedules large bags of tasks onto multiple cloud platforms. The individual tasks are scheduled in a self-scheduling manner onto the allocated machines. Aided by an initial sampling phase, BaTS computes a list of approximated Pareto solutions for scheduling the bag of tasks. This list of budget estimates provides the user with flexible control over budget and makespan. The list approximates the Pareto Front of solutions by using a genetic algorithm approach [20]. The execution phase allocates a number of machines from different clouds, and adapts the allocation regularly by acquiring and/or releasing machines in order to minimize the overall makespan while respecting the given budget limitation.

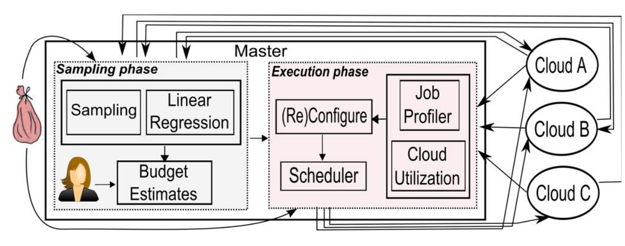

Fig. 2. BaTS sampling phase (left) and execution phase (right).

Our task model requires no prior knowledge about the task execution times; it only needs the total number of tasks. About the machines, we assume that they belong to certain categories (e.g. EC2's "Standard Large") and that all machines within a category are homogeneous. The only information BaTS uses about the machines is their (hourly) price.

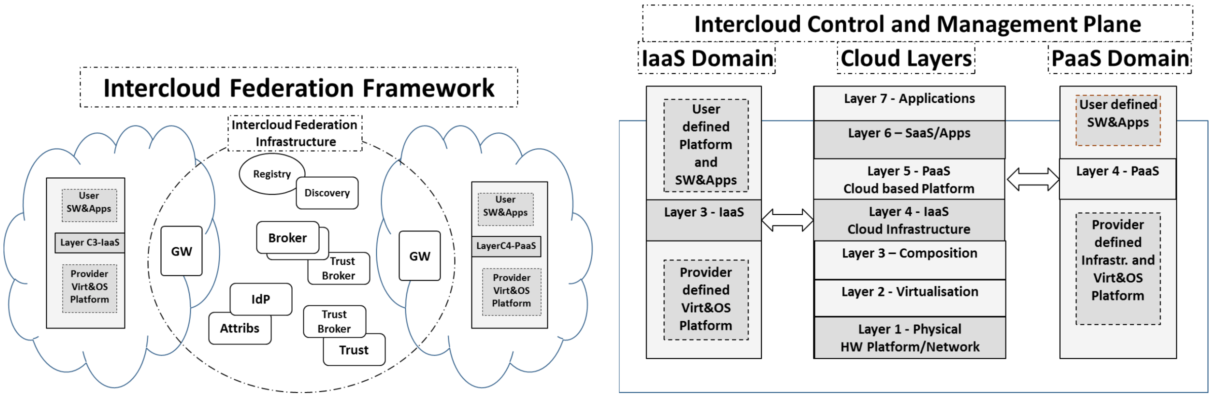

Fig. 1. The detailed architecture of two Intercloud Architecture Framework components: ICFF (left) and ICCMP (right).

Figure 2 sketches BaTS' overall system architecture. BaTS itself runs on a *master* machine, where the bag of tasks is available. Figure 2(left) sketches the sampling phase, where BaTS learns the bag's stochastic properties and uses linear regression to translate task completion times across clouds. BaTS generates a list of budget estimates accordingly, reflecting execution speed and profitability (price/performance ratio). The user is then asked to select one of the budgets (e.g., faster or cheaper) corresponding to a desired schedule. The user's choice then determines the machines allocated by BaTS for the execution phase, shown in Figure 2(right). Here, BaTS allocates machines from various clouds and lets the scheduler dispatch the tasks to the cloud machines. Feedback, both about task completion times and cloud utilization, is used to reconfigure the clouds periodically, as needed.

C. Pareto dominance, Pareto optimality and Pareto fronts

Formal definitions [21] of Pareto dominance, optimality and fronts abound. Intuitively, a problem that requires optimizing multiple objectives has several aspects: first, the input variables representing the available types of resources and their respective ranges. Secondly, the objectives, which are (possibly conflicting) functions of these variables. Thirdly, the feasibility space, that is the range of possible values for each objective function, given the ranges of the input variables.

The optimization takes place in the feasibility space, finding the best possible combinations of objective values, which will be mapped back to the corresponding combination of values of input variables (called a solution). The tough problem is deciding which are the best possible combinations, that is the best points in the feasibility space. Here, Pareto introduced the concept of dominance, which simply states that if a given solution S_1 outperforms another solution S_2 for at least one objective function, while it performs equally well for the remaining objective functions, then S_1 dominates S_2 (is strictly better). Based on this, a *Pareto optimal* solution is a non-dominated one (no other solution is strictly better). The set of all non-dominated solutions corresponds to a set of points in the feasibility space, called a *Pareto front* or *Pareto set*.

IV. USE CASES: RUNNING TASK FARM APPLICATIONS USING BATS ENHANCED BY ICAF COMPONENTS

Here, we investigate how the ICAF and BaTS systems should inter-operate to enable the execution in a multi-cloud environment of task farming applications. We look at different types of application execution requirements from a user's perspective and learn how they map to interoperability requirements.

A. Task Farming in a multi-cloud environment

We first analyze the interoperability issues encountered when deploying BaTS in a multi-cloud environment. We assume that a user has been running her bag-of-tasks application on cloud A (e.g. Amazon EC2 [22]) and that recently has decided to *also* use virtual machines from cloud B (e.g. BrightBox [2]). At a theoretical level, BaTS can already function in such a multi-cloud provider environment [23]. In practice, however, there are several aspects, functional and non-functional, that make the transition to a multi-cloud environment difficult.

From a functional perspective, the most difficult obstacle is the lack of a standardized set of specifications for the virtual resources. This leads to an increase both in terms of cost and time complexity of the BaTS' sampling phase, without any guarantees that the different types of virtual machines belonging to different cloud providers are not in essence redundant with respect to their perceived performance (the performance perceived by the application). To address this issue, we propose *investigating the use of semantic descriptions [24] to enable predicting similarities between virtual resource types.*

Another functional issue is the static nature of the cloud offerings. Though BaTS presents the user with a set of Pareto scheduling solutions, these are based on the current virtual resource types offered by cloud providers. Here, we propose *designing an ecosystem where the virtual resources are dynamically composed by the cloud providers to meet the specific needs of the application instance.*

From a non-functional perspective, we identify several aspects: how to ensure VM image compatibility? how to interact with the virtual compute resource (specific to the cloud provider API)? how to setup the network such that the BaTS' worker VMs can communicate with the BaTS' master/monitor? Indeed, the BaTS's modular design allows for new cloud adaptors to be easily integrated, albeit currently only to the extent where the virtual resource belongs to the same layer in the cloud stack, i.e. Infrastructure-as-a-Service. A simple extension implementing a generic RaaS cloud adaptor would suffice. Also, the image compatibility issue may be tackled, in the legacy-free cases, by approaches describing the image as a "recipe" [25].

Another non-functional issue is that the cloud providers interoperability is not necessarily symmetric (e.g. BrightBox provides Amazon EC2 metadata conversion, while Amazon EC2 is BrightBox-oblivious). At the same time, the cloud providers are constantly changing their interoperability policies. Indeed, the RaaS cloud paradigm would have to solve this issue; otherwise, the interoperability policy management would become the responsibility of the user application execution manager, i.e. the BaTS' Master.

An important security-related non-functional issue stems from the management of customer credentials as well as the customer-provider trust relationship. Here, the granularity at which the cloud providers offer their resources is very important for the design and performance of such authentication/authorization systems [26], [27].

B. ***Data-intensive*** *Task Farming*

Data-intensive task farming applications are a subcategory of the generic use-case presented in Section IV-A. Here, the focus is on the trade-off between moving the data to the computational resources or processing the data locally. We assume that company A has data stored in cloud A and company B has data store in cloud B; they need to cooperate at the analytics level to increase the Value dimension of their datasets.

One important functional interoperability issue is the analytics tools compatibility. Here, a set of semantic descriptions should be translated from the vocabulary of company A to that of company B and vice-versa.

When revisiting staging approaches applied in Grid environments [28], such as streaming instead of I/O operations, we identify a very important non-functional issue: providing controllable network quality, such that it can meet different application requirements specific to different execution scenarios. A straightforward example is deciding what the necessary network capacity is when transferring large data sets back and forth in a real-time fashion, while also being constrained by "network rental" costs.

Another very important non-functional issue is that of trust and legal issues. If data is to be moved from the cloud provider of company A that of company B, company A has to have a trust and legal clearance with cloud provider B. If data is to be processed locally and analytical tools are going to be migrated between companies, they each should trust each other's set of tools (both with respect to security issues and full disclosure of post processing results).

C. ***Energy-aware*** *Task Farming*

These applications may be compute- or data-intensive, but the main concern is the energy consumption. For instance, companies may have a marketing incentive (e.g. social responsibility and community programs) to project their image as being green computing consumers. Here, we encounter an additional functional issue, that is deciding which cloud providers offer the "greenest" mix of resources (e.g. datacenter, compute power and network).

D. *Scientific Web Services*

Recent research [29] conducted within the COMMIT project [1] tackles the problem of optimizing the orchestration of static web services by offering dynamic handling mechanisms of Grid resources. However, proper cloud-aware decision mechanisms are still to be defined, as the load and available quantity of resources are no longer meaningful metrics; new metrics may now include *profitability* - does the application receive the best performance for the money spent? or *eco-awareness* - does the application consume the least energy required to maintain the user-specified level of performance?

In this section, we presented several functional and non-functional aspects related to application execution in an Intercloud environment. Based on this summary, we argue that a new approach to virtual resource composition and scaling is required in the presence of Intercloud (computing).

V. The Intercloud Operation and Management Framework

The ICOMF role is to tackle new interoperability challenges at the application level from a user perspective. Usually, cloud customers will not know what the range of cloud offerings is and how these virtual resources map to their application performance. In the Intercloud context, the first question ICOMF should answer is *what* combination of virtual resources would provide the best match for a user's application according to the user requirements at that time. A second question, equally important, is *when* to assess migrating/adding another cloud provider to the set already in use by the application?

On the other side of the market, cloud providers do not know what virtual resource offerings would best suit each of their customers. The current solution has been to wrap these resources in offerings that *on average* are sufficiently well matched to application requirements.

The main research question behind ICOMF is *how* to decide the best virtual resource -to- application mapping in the presence of dynamic virtualisation granularity. A follow-up research question becomes *how* to decide on the right granularity of virtualization for each resource given a certain application and a certain cloud provider?

Generally, cloud providers offer different types of virtual resources with respect to their specification (memory, speed, number of cores); the providers may also differ in the non-functional aspects: the confidence interval associated with the SLA (how likely it is to perceive, on average, a certain performance) and/or the time granularity the resource is leased for (e.g. on-demand, reserved, spot).

[1] www.commit-nl.nl

The application instance may have different non-functional requirements from one execution to another, albeit the functional behavior on the same type of virtualized resources would not change. Moreover, depending on the application, the best solutions may comprise virtual resources from different layers (IaaS, PaaS, DaaS) and offered by different cloud providers.

A. The dynamic multi-cloud offering ecosystem

In our multi-cloud ecosystem, virtual resources are dynamically composed into offerings to meet application requirements. The main problem is finding the right granularity for each virtual resource type, such that the application performance requirements can be accurately modeled; the minimum level of granularity is given by the incurred overhead on the cloud provider's scheduling algorithms. Here, the heterogeneity of cloud providers' proprietary algorithms for resource virtualization may be addressed by maintaining a transparent approach. This means, each cloud provider considered for application execution in the sampling phase of BaTS may issue their own virtualization offering without disclosing details about their scheduling algorithms (e.g. hardware resource granularity, mechanisms for average perceived performance).

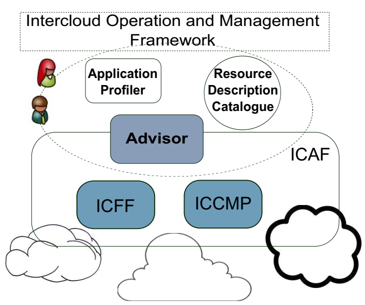

Fig. 3. The Intercloud Operation and Management Framework (ICOMF) components.

We depict the architecture of the ICOMF component in Figure 3. The decomposition of the application performance requirements needs information about the application execution that can be gathered during an initial sampling phase (similar to the one used by BaTS). Here, our Advisor component collects information about how efficient different virtualization granularities are with respect to the application at hand.

The Advisor component then constructs a Pareto set of (nearly-)optimal solutions that consider customized virtual resources with respect to the virtualization granularity.

The next step is to present this Pareto set to the cloud providers and verify which solutions can be met by their offerings. The Pareto set would contain solutions that consist of a number of units for each resource, which then the providers can package/virtualize accordingly. Here, the granularity of virtualization depends on the cloud provider's infrastructure, set of scheduling algorithms and business models.

Once the feasible Pareto set is constructed, the user may choose which solution best meets her requirements at that time. Here, the multi-level SLA management plays an important role, since it allows such flexible virtual resource packaging.

The question remains however of who should *own* the Advisor component. In our approach, the Advisor may coexist with any of the traditional owner types: the cloud provider, the cloud intermediary (operator) or the cloud customer. The challenge here is to provide sufficient guarantees to the user that the Pareto solutions are not biased by covert agreements between the party responsible for an Advisor implementation and one or more cloud providers and/or operators.

An important aspect to be considered is the trust relationship required between customers and providers, and between providers. This is an important aspect to be considered by future work.

The relationships between ICOMF and other ICAF components are described below:

a) ICOMF and ICCMP: The ICCMP will provide the essential lower level mechanisms for an Intercloud operation and management ecosystem, such as VM migration, object routing, inter-cloud configuration and monitoring. All these features are necessary for application execution in a multi-cloud environment.

b) ICOMF and ICFF: The ICFF will enable the vertical flexibility of virtual resource composition in an Intercloud context. Moreover, by supporting federation at the level of semantics and namespaces, the ICFF will alleviate the issue of a standardized virtual resource description. In turn, this will enable application profiling optimizations.

Compared to the approach to Intercloud presented by Bernstein [5], the ICAF differs with respect to several aspects: a) *addressing* follows the Geysers solution [30] by relying on the ICCMP component of ICAF; b) *presence and messaging* is expected to place a larger importance on the location services and it also relies on the ICCMP component. However, our work also adheres to the requirement that virtualized resources, especially virtual machines, should be portable within a multi-cloud environment. Next to that, we also stress the need for interoperable cloud services. We also identify a recent issue, related to a standardized set of specifications for the virtualized resources, as pursued by the efforts of the Distributed Management Task Force's (DMTF) initiative for Open Virtualization Format (OVF) [31].

VI. Service Delivery and Lifecycle Management

The on-demand cloud services provisioning requires a well-defined provisioning workflow and service lifecycle model. The ICAF Service Delivery Framework (SDF) extends the TeleManagement Forum SDF [32] with the necessary extensions to allow dynamic services provisioning, modification and recovery. The SDF combines in a provisioning workflow all processes that are run by different supporting systems and executed by different actors.

Figure 4 illustrates the main service provisioning or delivery stages that address specific requirements of the provisioned on-demand computational services:

- Service Request Stage (including SLA negotiation). The SLA can describe Quality of Service (QoS) and security requirements of the negotiated infrastructure service along with information that facilitates authentication of service requests from users. Here, we move towards the concept of dynamic SLA-rewriting and

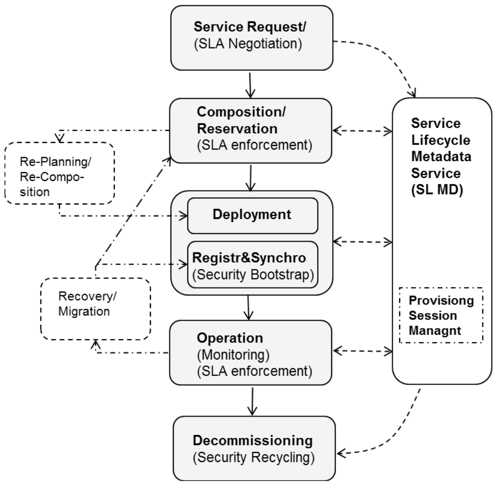

Fig. 4. SDF stages in on-demand (inter-)cloud services provisioning.

propose that this stage be under the supervision of the Advisor component, which would act on behalf of the user.

- Composition/Reservation Stage. This stage may require access control and SLA/policy enforcement. The Advisor component introduced in this paper provides the initiator of the on-demand cloud services with a relevant selection of provisioning options. The Advisor component will also ensure that the reservation and composition of resources follow the user-selected plan.
- Deployment Stage, including services Registration and Synchronisation. The deployment stage begins after all component resources have been reserved and includes distribution of the common composed service context (including security context).
- Registration and Synchronisation stage (which can be considered as optional) specifically targets scenarios with provisioned service migration or re-planning.
- Operation Stage (including Monitoring). This is the main operational stage of the provisioned on-demand cloud services. Monitoring is an important functionality of this stage to ensure service availability and secure operation, including SLA enforcement. Two additional (sub-)stages can be initiated from the Operation stage, based on the monitoring information about the running service or resources state:
  - Re-composition(Re-planning) Stage should allow for incremental infrastructure changes and will invoke the Advisor component to issue new relevant provisioning options.
  - Recovery/Migration Stage can be initiated by either the user or the provider. This process can use a special Service Lifecycle Metadata repository to initiate a full or partial resource re-synchronisation; it may also require re-composition.
- Decommissioning Stage ensures that all sessions are terminated, data is cleaned up, and session security context is recycled. The decommissioning stage can also provide information to or initiate service usage accounting.

The implementation of the proposed SDF requires a special Service Lifecycle Metadata Repository (MD SLC as shown on Figure 4) to support consistent services lifecycle management. MD SLC keeps the service's metadata that include at least service state, service properties, and service configuration information; the MD SLC is maintained by the cloud (federation) that offers the respective service(s). This repository may be consulted by the ICOMF when choosing possible resource candidates.

VII. Conclusion and Future Work

In previous work we have proposed an Intercloud Architecture Framework that would address generic issues of cloud interoperability. We have also designed and implemented a multi-cloud aware application scheduler, BaTS, that proposes relevant provisioning options and enforces application execution within a user-selected schedule.

In this paper, we have analyzed the requirements of an Intercloud Operation and Management Framework by studying several typical application use cases. Based on this requirements, we have proposed a new ecosystem for dynamic virtual resource composition. Our main contribution is that our model is user-centric and focuses on the specific application execution requirements, by leveraging incubating virtualization techniques. From a cloud provider perspective, the ecosystem provides more insight into how to best package the virtualized resources.

Our next step is to implement a prototype ecosystem and analyze its impact on real-world applications and cloud providers, such that it may be adopted by the industry.

VIII. Acknowledgment

This research was partially funded by the COMMIT project.

References

[1] "Google compute engine," https://cloud.google.com/products/compute-engine.

[2] "Brightbox," http://www.brightbox.com.

[3] "Rackspace," http://www.rackspace.com.

[4] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond, and M. Morrow, "Blueprint for the intercloud - protocols and formats for cloud computing interoperability," in *ICIW*, 2009, pp. 328–336.

[5] R. Buyya, R. Ranjan, and R. N. Calheiros, "Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services," in *ICA3PP (1)*, 2010, pp. 13–31.

[6] Y. Demchenko, M. Makkes, R. Strijkers, and C. de Laat, "Intercloud architecture for interoperability and integration," in *International Workshop on Emerging Issues in Cloud Computing (EIC) co-located with the 4th IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2012, pp. 666–674.

[7] Y. Demchenko, C. Ngo, C. de Laat, J. A. Garcia-Espin, S. Figuerola, J. Rodriguez, L. M. Contreras, G. Landi, and N. Ciulli, "Intercloud architecture framework for heterogeneous cloud based infrastructure services provisioning on-demand," *2013 27th International Conference on Advanced Information Networking and Applications Workshops*, pp. 777–784, 2013.

[8] L.-J. Zhang and Q. Zhou, “Ccoa: Cloud computing open architecture,” *2012 IEEE 19th International Conference on Web Services*, pp. 607–616, 2009.

[9] A. Takefusa, H. Nakada, R. Takano, T. Kudoh, and Y. Tanaka, “GridARS: A grid advanced resource management system framework for intercloud,” in *CloudCom*, 2011, pp. 705–710.

[10] P. Bosch, A. Duminuco, F. Pianese, and T. Wood, “Telco clouds and Virtual Telco: Consolidation, convergence, and beyond,” in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, 2011, pp. 982–988.

[11] R. Buyya, C. S. Yeo, and S. Venugopal, “Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities,” in *Proceedings of the 2008 10th IEEE International Conference on High Performance Computing and Communications*, ser. HPCC ’08, 2008.

[12] Y. Sun, T. Harmer, A. Stewart, and P. Wright, “Mapping application requirements to cloud resources,” in *Euro-Par 2011: Parallel Processing Workshops*.

[13] O. Agmon Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, “The Resource-as-a-Service (RaaS) cloud,” in *USENIX Conference on Hot Topics in Cloud Computing (HotCloud)*, 2012.

[14] “Apache Cloudstack,” https://cwiki.apache.org/confluence/display /CLOUDSTACK/Dynamic+scaling+of+CPU+and+RAM.

[15] J. A. Garcia-Espin, J. F. Riera, S. Figuerola, and E. López, “A multi-tenancy model based on resource capabilities and ownership for infrastructure management,” in *CloudCom*, 2012.

[16] “Generalised architecture for dynamic infrastructure services (GEYSERS project).” http://www.geysers.eu.

[17] J. Happe, W. Theilmann, A. Edmonds, and K. Kearney, “A reference architecture for multi-level sla management,” in *Service Level Agreements for Cloud Computing*. Springer New York, 2011, pp. 13–26.

[18] A.-M. Oprescu, T. Kielmann, and H. Leahu, “Budget estimation and control for bag-of-tasks scheduling in clouds,” *Parallel Processing Letters*, vol. 21, no. 2, pp. 219–243, 2011.

[19] A. Vintila, A.-M. Oprescu, and T. Kielmann, “Fast (re-)configuration of mixed on-demand and spot instance pools for high-throughput computing,” in *International Workshop on Optimization techniques for Resources Management in Clouds (ORMaCloud) co-located with the 22nd ACM Symposium on High-Performance Parallel and Distributed Computing*, 2013, pp. 25–32.

[20] M. Ehrgott, *Multicriteria Optimization (2. ed.)*. Springer, 2005.

[21] Amazon Web Services, http://aws.amazon.com.

[22] A.-M. Oprescu, *Stochastic Approaches to Self-adaptive Application Execution on Clouds*, ser. Doctoral Dissertation. Amsterdam: Vrije Universiteit, 2013.

[23] M. Ghijsen, J. van der Ham, P. Grosso, C. Dumitru, H. Zhu, Z. Zhao, and C. de Laat, “A semantic-web approach for modeling computing infrastructures,” *To appear in Computers and Electrical Engineering*, 2013.

[24] Y. YAO, “A scheme for defining and deploying trusted virtual machines to grid sites using configuration management systems,” in *HEPiX Workshop*, 2010.

[25] C. Ngo, P. Membrey, Y. Demchenko, and C. de Laat, “Policy and context management in dynamically provisioned access control service for virtualized cloud infrastructures.” in *ARES*, 2012, pp. 343–349.

[26] Y. Demchenko, C. de Laat, and C. Ngo, “Toward a dynamic trust establishment approach for multi-provider intercloud environment,” in *Proceedings of the 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom)*, 2012, pp. 532–538.

[27] V. Korkhov, D. Vasyunin, A. Wibisono, V. Guevara-Masis, A. Belloum, C. de Laat, P. Adriaans, and L. Hertzberger, “Ws-vlam: towards a scalable workflow system on the grid,” in *Proceedings of the 2nd workshop on Workflows in support of large-scale science*, ser. WORKS ’07. ACM, 2007, pp. 63–68.

[28] R. Cushing, S. Koulouzis, A. Belloum, and M. Bubak, “Dynamic handling for cooperating scientific web services,” in *E-Science (e-Science), 2011 IEEE 7th International Conference on*, 2011.

[29] “RFC 3945. generalized multi-protocol label switching (gmpls) architecture.” http://www.ietf.org/rfc/rfc3945.txt.

[30] Distributed Management Task Force, http://dmtf.org/standards/ovf.

[31] “TR139, Service Delivery Framework (SDF) Overview, Release 2.0.” http://www.tmforum.org/TechnicalReports/ TR139ServiceDelivery/34303/article.html.