OPTIMIS: a Holistic Approach to Cloud Service Provisioning

Ana Juan Ferrer
Atos Origin, Barcelona, Spain
ana.juanf@atosresearch.eu

Francisco Hernández, Johan Tordsson, Erik Elmroth
Dept. Computing Science, Umeå University, Sweden
{hernandf, tordsson, elmroth}@cs.umu.se

Csilla Zsigri
The 451 Group, Barcelona, Spain
csilla.zsigri@the451group.com

Raül Sirvent, Jordi Guitart, Rosa M. Badia
Barcelona Supercomputing Center
Barcelona, Spain
{Raul.Sirvent, jordi.guitart, rosa.m.badia}@bsc.es

Karim Djemame
School of Computing
University of Leeds, UK
karim@comp.leeds.ac.uk

Wolfgang Ziegler
Fraunhofer – SCAI
Sankt Augustin, Germany
wolfgang.ziegler@scai.fraunhofer.de

Theo Dimitrakos, Srijith K. Nair
British Telecom, London, UK
{theo.dimitrakos, srijith.nair}@bt.com

George Kousiouris, Kleopatra Konstanteli, Theodora Varvarigou
National Technical University of Athens, Greece
{gkousiou, kkonst, dora}@telecom.ntua.gr

Benoit Hudzia
SAP Research Belfast, United Kingdom
Email: benoit.hudzia@sap.com

Alexander Kipp, Stefan Wesner
High Performance Computing Center Stuttgart, Germany
{kipp, wesner}@hlrs.de

Marcelo Corrales, Nikolaus Forgó
Institut für Rechtsinformatik, Leibniz Universität Hannover, Germany
{corrales, nikolaus.forgo}@iri.uni-hannover.de

Tabassum Sharif, Craig Sheridan
Flexiant Limited, Livingston, UK
{tsharif, csheridan}@flexiant.com

***Abstract*—We present the fundamentals for a toolkit for scalable and dependable service platforms and architectures that enable flexible and dynamic provisioning of cloud services. The innovations behind the toolkit are aimed at optimizing the whole service life cycle, including service construction, deployment, and operation, on a basis of aspects such as trust, risk, eco-efficiency and cost. Notably, adaptive self-preservation is crucial to meet predicted and unforeseen changes in resource requirements. By addressing the whole service life cycle, taking into account the multitude of future cloud architectures, and a by taking a holistic approach to sustainable service provisioning, the toolkit is aimed to provide a foundation for a reliable, sustainable, and trustful cloud computing industry.**

I. Introduction

Contemporary cloud computing solutions, both research projects and commercial products, have mainly focused on providing functionalities at levels close to the infrastructure, e.g., improved performance for virtualization of compute, storage, and network resources, as well as necessary fundamental functionality such as virtual machine (VM) migrations and server consolidation. In the cases when higher-level concerns are considered, existing solutions tend to focus on functional aspects only. Furthermore, existing Platform as a Service (PaaS) environments are typically offered through proprietary APIs and limited to a single infrastructure provider. In order to move from a basic cloud service infrastructure to an improved cloud service ecosystem, there is a great need for tools that support higher-level concerns and non-functional aspects in a comprehensive manner.

In this work we focus on five higher-level concerns that in our view must be addressed for a wider adoption of cloud computing:

1) Service life cycle optimization
2) Dependable sociability = Trust + Risk + Eco + Cost
3) Adaptive self-preservation
4) Multi-cloud architectures
5) Market and legislative issues

Notably, these five concerns cover most of the ten key obstacles to growth of cloud computing identified in a recent report [3]. In addressing these concerns, we focus on a holistic approach to cloud service provisioning and argue that a single abstraction for multiple coexisting cloud architectures can enable the next generation cloud service ecosystem. The outcome of our holistic approach is the *OPTIMIS Toolkit*. We present the design of the toolkit and discuss how it addresses the higher-level concerns introduce above.

Our work is based on the assumption that clouds will be available as private and public, that they will be used in isolation or in a variety of conceptually different combinations, and that they will be internal or external to individual organizations or cross-organizational consortia.

The main stakeholders throughout this work are service providers and infrastructure providers, although it can be

foreseen that our results can also impact actors such as brokers, and service consumers (end-users). Henceforth, we consider the following definitions for the service and infrastructure provider roles:

- *Service Providers (SPs)* offer economically efficient services with assessed and guaranteed environmental impact using hardware resources provided by infrastructure providers. The services are directly accessed by end-users or orchestrated by other SPs.
- *Infrastructure Providers (IPs)* offer infrastructure resources required for hosting of services. Their goal is to maximize their profit from tenants by making efficient use of the infrastructures, and possibly by outsourcing partial workloads to partnering providers.

The element of interaction between SPs and IPs is a service. Notably, SPs and IPs have conflicting economical and performance goals that result in interesting problems in cases where they are both part of the same organization. It is foreseen that both types of providers are in need of more feature-rich analysis and management tools in order to provide economically and ecologically sustainable services throughout the whole service life cycle.

The outline of the paper is as follows. Sections II to VI present five higher-level concerns for future clouds. These concerns guide the design of the OPTIMIS Toolkit that is presented in Section VII, including a high-level view of its components and an illustration on how the toolkit can be used to create various cloud architectures. Finally, we share our concluding remarks in Section VIII.

II. Service Life Cycle Optimization

There are three fundamental steps in the service life cycle, construction of the service, deployment of the service to an IP, and the operation of the service using the IP resources.

A. Service Construction

In the service construction phase, the SP first builds (i.e., implements, assembles, and/or orchestrates) the service and then prepares it for deployment and operation on the IP. The activities performed include preparation and configuration of images for the VMs that constitute the service as well as specification of dependencies among the different service components. This process can be facilitated by a general-purpose programming model that simplify the service construction as well as by an expressive language for SLAs that allows a wide range of configuration parameters related to dependable sociability, to be incorporated in service construction.

Currently, there is no programming model specifically tailored for clouds. On the one hand developers are limited to use application-specific platforms [16], restrictive computing paradigms [10], [35], or platforms for a single cloud middleware [28]. A common way of offering these solutions is by wrapping them as a PaaS environment or even by offering a proprietary API for a particular middleware. On the other hand, developing high-level services from raw infrastructure through use of IaaS is a manual and ad-hoc process, hindering broader cloud adoption as service development becomes expensive and time consuming.

The challenge of service construction resides in designing and developing easy ways to create complex services. To this end, applications need to be abstracted from their execution environment and the development of new services, including those composed from adapting and combining legacy- and licensed software, must be facilitated. For the latter, novel license management technologies are required that significantly extend currently available solutions for management of license tokens in distributed environments [25]. The composition of services as a mix of software developed in-house, existing third-party services, and licence-protected software is a clear contrast to commonly used approaches for service composition [19].

B. Service Deployment

In the service deployment phase, the service is placed on an IP for operation. The main objective during this phase for the SP is to select the most suitable IP for hosting a service, including negotiation of SLA terms. Another task is to propagate the contextualization information required for instantiating the service once deployed. On the IP side, admission control is based on an evaluation of the benefit from the additional service. Contemporary cloud SLA mechanisms [2], [8] are typically limited to cost-performance tradeoffs. For example, it is not possible to automatically evaluate levels of trust and risk, or to negotiate use of license-protected software. Furthermore, existing deployment tools are limited to use of single clouds as differences in contextualization mechanisms [21], [22] hinder multi-cloud deployment.

To overcome current limitations, deployment optimization tools are needed that support deployment given a set of policies and that allows SPs to specify required SLA terms for the service. The policies governing the deployment include the degree of trust expected from a provider, the level of risk with regard to cost thresholds, energy consumption limits, performance levels, etc.

C. Service Execution

Service execution is the last phase in the service life cycle and includes two different but related procedures, performed by the SPs and IPs. The overall objectives of these stakeholders differ and as a result, there is a conflict of interest in the performed management tasks. On one hand, the SP performs a set of management operations in order to meet the high-level Business Level Objectives (BLOs) specified during service construction. These include, for instance, constant monitoring of service status and triggering actions to increase and decrease capacity to adhere to SLAs, i.e., enact elasticity rules, and mechanisms for monitoring and continuous assessment of the risk level of IPs in order to apply the corresponding corrective actions. On the other hand, IPs perform autonomic actions to, e.g., consolidate and redistribute service workloads, replicate and redistribute data sets, etc. with the overall goal of achieving the most efficient use of the provider infrastructure

and hence maximize its own objectives, potentially at the expense of the goals of the SPs.

Contemporary tools for service execution optimization focus on mechanisms for monitoring service status and for triggering capacity variations to meet elasticity requirements [32]. These tools tend to use only SLAs and infrastructure status for making decisions and either neglect business-level parameters such as risk, trust, reliability, and eco-efficiency, or consider them in isolation. For instance, eco-efficient policies for the operation of hosting centers aiming to minimize its power consumption have been investigated [5], [6]. Similarly, trust mechanisms have been studied in the context of Grid resource selection in order to choose providers that are likely to provide better service according to their reputation [1]. In the same way, risk information has been used for decision making [11]. Finally, some resource management proposals for data centers and e-commerce systems are driven by business objectives or incorporate business level parameters in their management policies [4], [15], [27], though most of them target revenue as the only objective.

According to this, SPs and IPs require software components that in addition to the traditional performance indicators also take into account business-level parameters (e.g., risk, trust, reliability, etc.) in order to make decisions in a synergistic fashion that contribute to the overall provider goals. In order to achieve this type of decision making process, all management activities must be harmonized through the use of cloud governance processes that integrate all service requirements, from high-level BLOs to infrastructure requirements.

III. Dependable Sociability = Trust + Risk + Eco + Cost

Traditionally, relationships between stakeholders have been focused on cost-performance trade-offs. However, these economical factors are not enough for an open and highly dynamic environment in which relationships are created in an on-off basis with a possible high degree of anonymity between stakeholders. Instead, a broader perspective that also incorporates quality factors is required. Thus, on the one hand, it is necessary to offer methods and tools to quantitatively assess and evaluate stakeholders, e.g., through audit and monitoring functions including analysis of probability of service failure, risk of data loss, and other types of SLA violations. On the other hand, methods to measure stakeholder satisfaction are also important, e.g., individual and group perceptions, reputation of stakeholders regarding ecological aspects, or previous experiences. Altogether, these mechanisms confirm the dependability and reliability among members of a cloud ecosystem.

A. Trust – Reputation Management

Trust is a multifaceted aspect not only related to risk and security aspects, but also to perceptions and previous experiences. Selection of an IP depends on the trust that it will provision the service correctly and securely. Conversely, for an IP, knowing a customer's reputation improves the admission control evaluation, reducing the risk of breaking the economical or ecological goals of the IP.

Trust is often calculated by reputation mechanisms [20]. A reputation is a subjective measure of the perception that members of a social network has of one another. This perception is based on past experiences. The reputation ranks aggregate experiences of all members of the social network – in this case the social network is the cloud ecosystem, i.e., the combination of SPs and IPs. To create a comprehensive trustworthy system, the relationships that must be considered are of the types SP–IP and IP–IP. Trust estimations are determined from the trust rank of the SP or IP in other members of its own social network (cloud ecosystem) according to a transitive trust chain. The reputation mechanism must deliver trust measurements at two levels. For IPs, trust reflects their performance and the ability to accomplish promised levels of service. For SPs, trust assessment mechanisms relevant for establishing successful business networks include methods to identify SPs in long term relationships and to analyze SPs' historical behavior that can help to improve IP's management operations by e.g., prediction of future capacity. Tools are also required to determine the integrity of data disclosed by the ecosystem members as well as mechanisms to act accordingly, e.g., to blacklist dishonest providers,

B. Risk Assessment

Underpinning a successful cloud infrastructure is delivering the required QoS levels to its users in a way that minimizes risk, which is measured in terms of a combination of the likelihood of an event and its impact on the provision of a functionality. Risk is a cross-cutting concern for clouds that integrates factors such as trust, security, energy consumption, and cost. Earlier work in risk management for distributed systems has mainly focused on operational aspects such as failures and performance degradation, and assumed a very IaaS centric view under a specific resource reservation model [12]. There is a need for tools for the definition, assessment and management of risk based on variations in levels of the proposed eco-factors for both stakeholders: SPs during service construction, deployment, and operation; and IPs during admission control and internal operations. Such risk management mechanisms for cloud services, which consider inherent aspects of clouds such as energy consumption, the cost of reconfiguration and migration, and the reliability and dependability of the provided services, will maintain secure, cost-effective, and energy-efficient operations.

C. Green Assessment

Environmental concerns reflected in upcoming legislation have increased the awareness of the ecological impact of the ICT industry. The result is that the level of ecological awareness can now be a deciding factor between competing providers. However, environmental concerns are not the only reason for the growing interest in green data centers, rising electricity prices can also guide the deployment of services to locations in which they are provisioned in a more efficient

way. The consequence is that IPs must now focus more than ever on improving their energy efficiency.

Cloud computing in itself contributes to reduce power consumption by consolidating workloads from different customers in a smaller number of physical nodes, turning off unused nodes [26]. However, a difficult aspect to handle is the tradeoff between performance and power consumption [7], [23]. To address this tradeoff, energy efficiency must be treated as the other critical operating parameters, already including service availability, reliability, and performance. The aim is to minimize power consumption while still fulfilling the BLOs of the IP.

A solution requires a broad set of mechanisms, including tools for logging and continually assessing the ecological impact at the service level, as well as theoretical models to characterize the power consumption of services depending on configuration parameters (e.g., clock frequency, resource usage, and number of threads used). These mechanisms enable the prediction of future energy impact based on run-time state, historical usage patterns, and estimates of future demands. Finally, actuators are required to carry out all eco-efficiency decisions during resource provisioning, data placement, and service placement.

D. Cost and Economical Sustainability

Fulfilling high trust levels between stakeholders, reduced risk, and eco-efficient provisioning is trivial if cost is not an issue. However, economical aspects are necessary to balance the previous three goals, and cost must be an explicit parameter throughout the full service life cycle. Current commercial providers offer a variety of capabilities under different pricing schemes, but it is hard to differentiate among the offerings without sufficient knowledge of the repercussions on internal performance, ecologic, and economic goals. To improve this situation, more complex economic models are needed. These models must include features to compare economical terms between alternative configurations. To this end, such models must employ business related terms that can be translated to service and infrastructure parameters during development and deployment of services. During the operation of a service, it is necessary to optimize economical factors through a combination of runtime monitoring, analysis of historical usage patterns, and predictions of future events. The latter helps to anticipate future service economic trends. All of these actions create an economic policy framework in which the stakeholders can specify the autonomic behavior they expect from the elements under their management responsibility.

IV. Adaptive Self-Preservation

Service and infrastructure management in clouds is difficult due to their ever-growing complexity and inherent variability in environmental conditions. Quick responses to these variations are necessary to fulfill the agreed SLAs. Accordingly, human administration becomes unfeasible and building self-managed systems seems to be the only way to succeed.

Although self-management for cloud infrastructures is a novel area of research, many of the proposals for virtualized hosting centers that allow automated adjustment of resource allocation could be applied [30], [33]. However, many of the previously proposed solutions have two main limitations. First, they typically exhibit a lack of expressiveness in self-management due to a lack of a holistic view of management. This results in management actions, e.g., resource allocation, monitoring, or data placement, that are performed in isolation and are as such not optimal. Second, existing solutions tend to use only SLAs and infrastructure status for making decisions, neglecting business-level parameters such as risk, trust, reliability, and eco-efficiency, or considering them in isolation, as discussed in Section II-C.

To overcome this problem, SPs and IPs require that all management actions are harmonized by overarching policies that incorporate, balance, and synergize aspects of risk assessment, trust management, eco-efficiency, as well as economic plausibility.

The management actions must be handled by software components able to monitor and asses their own status and adapt their behavior to ever changing conditions. Aspects to consider in this decision are overall BLOs, infrastructure capabilities, historical usage patterns, and predictions of future demands. The result is an integrated solution capable of a wide range of autonomic management tasks [24] including self-configuration, i.e., automatic configuration of components, self-healing, i.e., automatic discovery and correction of faults, and self-optimization, i.e., automatic optimization of resource allotments and data placement. Example autonomic management tasks include SLA enforcement, recovery of service operation upon resource failure, VM placement optimization (including migration), enactment of elasticity policies (vertical and horizontal scalability), consolidation of services to improve eco-efficiency, management of advance reservations, and data replication for fault tolerance or performance improvements.

V. Multi-Cloud Architectures

There are at least two fundamentally different architectural models for cloud service provisioning using multiple external clouds:

- In the *federated cloud* case, an IP can sub-contract capacity from other providers as well as offer spare capacity to a federation of IPs. Parts of a service can be placed on remote providers for improved elasticity and fault tolerance, but the initial IP is solely responsible for guaranteeing the agreed upon SLA (with respect to performance, cost, eco-efficiency, etc.) to the SP.
- In a *multi-provider hosting* scenario, the SP is responsible for the multi-cloud provisioning of the services. Thus, the SP contacts the possible IPs, negotiates terms of use, deploys services, monitors their operation, and potentially migrates services (or parts thereof) from misbehaving IPs. IPs are managed independently and placement on

different providers is treated as multiple instances of deployment.

Each model has benefits and drawbacks. However, to date, these models have only been studied in isolation [22], [32], [34], which essentially creates either-or situations. Instead, for a more flexible provisioning model, it is important to be able to use multiple clouds without distinguishing whether a service is hosted within a single cloud or across multiple providers, i.e., clouds must be able to be combined into arbitrary, hierarchical architectures. To this end, it is imperative to create a single abstraction without regard of architectural style. In order to accomplish this, there are a number of challenges that must be solved, including: verification of SLA adherence; metering, accounting and billing of services running out of a provider's boundaries; managing software license authorizations, particularly when migrating a service to different providers; replication, synchronization, and backup of data between providers; evaluation of economical efficiency associated with using external providers; and establishing an inter-cloud security context for governing all interactions between clouds.

VI. Market and Legislative Issues

Clouds bring change to user behavior. The focus of attention is moving away from how a service is implemented or hosted to what the service offers, a shift from buying tools that enable a functionality to contracting third-party services that deliver this functionality on demand in a pay-per-use model [14]. There is a massive surge in interest around private and hybrid clouds. With new application use cases emerging on a regular basis, numerous commercial on-ramps are seeking to provide access to multiple clouds, and startups and incumbent providers alike are targeting cloud service brokerage. These changes in the landscape create opportunities for new roles, relationships, and value activities but also create additional concerns related to legal compliance.

It is important to assess from the very beginning those associated legal risks in cloud computing and create a framework for minimizing or mitigating those risks, particularly when presupposing that data moves geographically. In such cases, data protection and privacy, being issues of cross-border jurisdictional nature as they concern the acquisition, location, and transfer of data [17], are important and call for a data protection framework and security infrastructure [9], [18]. Furthermore, legally and non-legally binding guidelines concerning green IT strategies and legislative and jurisdictional issues are key to infrastructure and service providers when it comes to decision making [13], [29]. In addition, intellectual property and contractual issues concerning ownership and rights in information and services located in the cloud need to be tailored and taken into account when designing a cloud computing toolkit [31].

VII. The OPTIMIS Toolkit

Our response to the challenges presented in the previous section is the OPTIMIS Toolkit, currently under development.

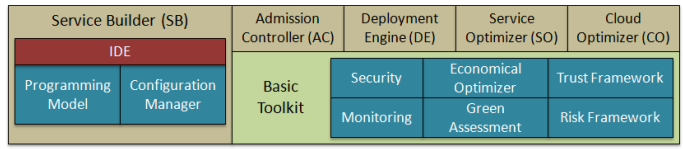

Fig. 1. High level view of the components in the OPTIMIS Toolkit. The Basic Toolkit addresses quantitative and qualitative analysis that help in making optimal decisions regardless of the invoking component. Organizations can act as SPs and/or IPs depending on the components that they chose to adopt (from the Admission Controller, Deployment Optimizer, Service Optimizer, and Cloud Optimizer).

The toolkit consists of a set of fundamental components realizing an anticipating a variety of architectures for simultaneous use of multiple clouds. Figure 1 illustrates the high-level components of the toolkit: the *Service Builder*, the *Basic Toolkit*, the *Admission Controller (AC)*, the *Deployment Engine (DE)*, the *Service Optimizer (SO)*, and the *Cloud Optimizer (CO)*.

The Service Builder component is used during the service construction phase and enables developed services to be delivered as Software as a Service (SaaS). A service programmer has access to an integrated development environment that simplifies both development and configuration of the service, two important actions for efficient utilization of cloud resources. In our novel programming model for service development, a service is a collection of *core elements*, e.g., services built from directly from source code, existing services, licensed software, and legacy-software not developed specifically for clouds, as well as a set of dependencies between the core elements. During operation of the service, the core elements are orchestrated by a runtime environment that analyzes the dependencies defined during service construction.

Each core element has a set of functional and non-functional requirements associated, e.g., requested performance, amount of physical memory and CPU characteristics, response time, elasticity aspects, service and security level policies, ecological profile, etc. In addition, there are also requirements among the core elements and between the service and its potential users. The requirements are encoded in a service manifest that is the input to the various toolkit components that handle service deployment and execution.

The Basic Toolkit provides functionalities common to components that are used during service deployment and execution. Some of the functionalities address the quantitative and qualitative requirements that we in Section III summarize under the term Dependable Sociability, whereas monitoring and security are functionalities that must be considered during several stages of the service life cycle. These are general purpose functionalities and evaluate similar aspects of management. However, component behavior is customized depending on the invoking module. This customization is fulfilled through the use of internal policies that adapt the decision making processes, e.g., based on the invoking component and the current stage of the service life cycle.

We now illustrate how the OPTIMIS Toolkit is used during deployment of services. For simplicity in understanding how

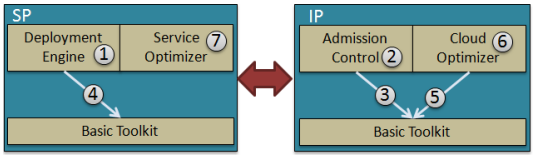

Fig. 2. Service deployment scenario that illustrates the interaction between the high-level components of the SP and IP.

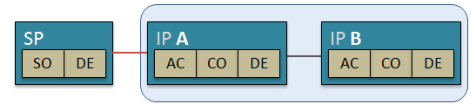

Fig. 3. Federated cloud architecture where the SP establishes a contract with IP A that is a member of the federation that includes IP B. The service is delivered using resources of either IP, or both.

the components interact, the scenario presented in Figure 2 is that of an IP delivering capacity to a SP. More complex scenarios can also be realized as described later in the section.

During service deployment the SP finds, by use of the DE, the best possible location for operation of the service. Prior to deployment, the DE must package the services and software stack into a set of VM images. In addition to the service images, the DE uses a deployment manifest specifying the functional and non-functional requirements of the service. The first step of deployment is negotiation between the DE and the IPs in order to find suitable candidates for service operation (*Step 1*).

An IP receiving a deployment request performs a probabilistic admission control to decide whether to admit the new service or not (*Step 2*). This test balances revenue maximization lead by business goals against penalties for misbehavior, i.e., from breaking SLAs of current running services. Example policies include *over-provisioning* (overbooking for revenue optimization) as well as *under-provisioning* (reserving capacity to minimize the risk of failures). The test is carried out by the Admission Controller (AC) component through use of the Basic Toolkit (*Step 3*). An integral part of the test is workload analysis of the current infrastructure and the new service, as well as capacity planning.

Using the Basic Toolkit, the DE evaluates all IPs that offer to run the service in order to chose an initial one (*Step 4*). This analysis is carried out considering both qualitative and quantitative factors as explained in Section III. After selecting an initial IP, the DE prepares the service images for deployment at the chosen IP. This includes adaptation to the used contextualization mechanism, as this can vary between providers.

In the IP, the process of accepting a new service starts by allocating space for the VMs and selecting their initial placement. The latter is a complicated process as placement must consider the predicted elasticity of the service and the non-functional constraints specified in the deployment manifest. Some of these constraints can even have legal ramifications regarding e.g., data protection and privacy or environmental guidelines as discussed in Section VI. This process is performed by the CO using the functionalities in the Basic Toolkit (*Step 5*). Data management, including transfer of VM images and other data required by the service, is performed prior to launching the VMs, whereas contextualization is performed after the VMs have been initialized (*Step 6*). The SO in the SP is notified (*Step 7*) once the deployment process completes.

The SO and CO also perform repeated management decisions during service operation, the SO on behalf of the SP and the CO for the IP. The SO continuously checks that the IP provisions the service according to the agreed SLAs, otherwise the SO can migrate the service to a different IP. On the other hand, the CO optimizes the IP's infrastructure resources. This includes, for instance, monitoring of infrastructure status, recovery from failures (e.g. by using checkpoints), and mechanisms for optimizing power consumption (e.g. by consolidating and migrating VMs), while at the same time protecting SLAs with SPs to avoid penalties and preserve reputation. The SO and CO both utilize the Basic Toolkit as the basis for their quantitative and qualitative analysis.

A. Flexible multi-cloud architectures

The combination of the five main components of the OPTIMIS Toolkit and their implementation by SPs and IPs, gives rise to a number of plausible multi-cloud scenarios where resources from more than one IP can be combined in novel ways. Some example compositions follow.

a) Federated architecture: In this scenario (Figure 3), several IPs (A and B) use the OPTIMIS Toolkit to establish a cooperation in which any IP can lease capacity from the other. The cooperation is carried out according to internal IP business policies. The SP is unaware of this federation as its contract is with a single IP (in this case IP A). However, the SP can indirectly pose constraints to the IPs in the federation through the use of non-functional requirements such as affinity of service components or juridical restrictions such as prevented data movement across country borders. The contracted provider (IP A) is fully responsible towards the SP even in the case of subcontracting of resources from the federation.

b) Multi-cloud architecture: In this scenario (Figure 4), the SP is responsible for the multi-cloud aspect of service operation. If IP A does not fulfill the agreed objectives, the SP can cancel the contract and move the service to a different IP (IP B). Notably, the SP is responsible both for negotiating with each IP and for monitoring the IPs during service operation. In more complex variations of this scenario, parts of the service can be hosted on multiple providers. By using APIs and adapters externally to the OPTIMIS components, the toolkit can also achieve interoperability with non-OPTIMIS providers (IP C in Figure 4). However, in such cases, the SP has to resort to less feature-rich management capabilities, and the risk levels for service provisioning increase accordingly.

c) Aggregation of resources by a third party broker: This scenario (Figure 5), introduces a new stakeholder, the broker, which aggregates resources from multiple IPs and offers these

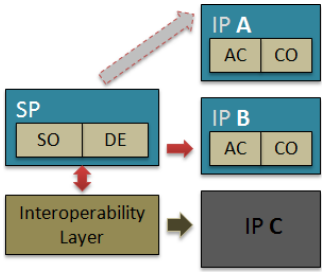

Fig. 4. Multi-cloud architecture in which the SP breaks the contract with IP A and re-deploys the service to IP B. The SP can also use infrastructure from an IP (C) that does not implement the OPTIMIS Toolkit. In this case the SP uses an interoperability layer that is external to the OPTIMIS components.

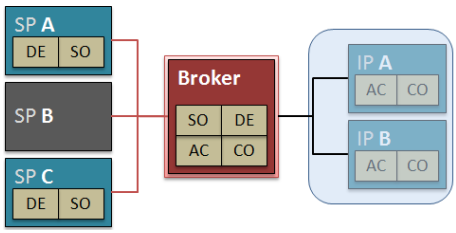

Fig. 5. In the brokering architecture, a third party broker aggregates resources from several IPs (A and B) and offer these resources to SPs.

to SPs. The broker thus acts as a SP to IPs and as an IP to SPs. Given the conflicting goals of these respective providers, there are many interesting concerns regarding the independence, honesty, and integrity of the broker. Benefits with this model for SPs include simplicity and potential cost reductions, as the broker can provide a single entry point to multiple IPs and may offer better prices due to bulk discounts from IPs. Management is simplified for an IP that offers capacity to a broker as the number of customers is decreased. Accordingly, trust and risk become easier to predict as the IP is likely to have fewer and longer term contracts with brokers, instead of a multitude of short interactions with potentially unknown SPs.

d) Hybrid cloud architecture: In this scenario (Figure 6), any organization that operates a private cloud is able to externalize workloads to public IPs. This is accomplished by monitoring the normal operation of the private cloud, through the CO component, and use capacity from public clouds (IPs A, B, and C, the former two in federation) when the local infrastructure is insufficient. The implementation of this scenario is significantly simplified once an organization is using the OPTIMIS Toolkit for managing its private cloud. Furthermore, this scenario can be extended if the organization makes use of the AC. In this case, the organization will be able to offer capacity from its private cloud to others (SP A) when that capacity is not needed for internal operations.

VIII. Concluding Remarks

We present the fundamentals of an on-going effort to develop the OPTIMIS Toolkit that supports the construction of multiple coexisting cloud architectures that make up the next generation cloud service ecosystem. With this toolkit, our goal is to provide a holistic solution to cloud service provisioning

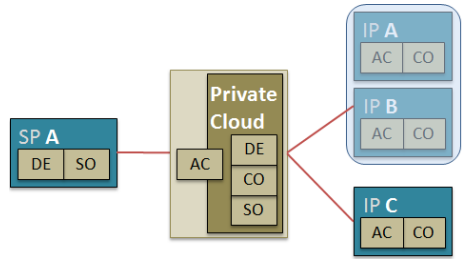

Fig. 6. Hybrid cloud architecture. An organization moves part of its operation to external providers (the federation formed by IPs A, B, and C). The organization can also sell capacity to the SP during periods of low load.

through a set of independent components that can be used to deliver advanced services.

The focus of the toolkit is on cloud infrastructure and service optimization throughout the service life cycle: construction, deployment, and operation of services. In the OPTIMIS Toolkit all management actions will be harmonized by overarching policies that consider trust and risk assessment to comply with economical and ecological objectives without compromising operational efficiencies. Assessing risk of economical and ecological parameters is a unique, albeit challenging, goal. Governance processes and policies will be defined to harmonize management activities throughout the service life cycle.

The self-service tools will enable developers to enhance services with non-functional requirements regarding allocation of data and VMs, as well as aspects related to performance (elasticity), energy consumption, risk, cost, and trust. The OPTIMIS Toolkit incorporates risk aspects in all phases of the service life cycle and uses trust assessment tools to improve decision making in the matching of SPs and IPs. Furthermore, the ecological impact of service provisioning is integrated in all relevant decision making. The toolkit also ensures that the desired levels of risk, trust, or eco-efficiency are balanced against cost, to avoid solutions that are unacceptable from an economical perspective. The OPTIMIS tools are aimed to enable SPs and IPs to perform monitoring and automated management of services and infrastructures, so as to compare different alternative configurations in terms of business efficiency. Notably, legislative and regulatory aspects are also incorporated in the toolkit, e.g., to address adoption challenges from regulatory and standards compliance requirements such as data privacy legislation.

The flexible and adaptable OPTIMIS Toolkit will enable and simplify the creation of a variety of provisioning models for cloud computing, including cloud bursting, multi-cloud provisioning, and federation of clouds. The provisioning on multi-clouds architectures and federated cloud providers will facilitate novel and complex composition of clouds that considerably extend the limited support for utilizing resources from multiple providers in a transparent, interoperable, and architecture independent fashion.

ACKNOWLEDGMENTS

We acknowledge the anonymous reviewers for their insightful feedback. Financial support for this work is provided by the European Commission's Seventh Framework Programme ([FP7/2001-2013]) under grant agreement number 257115, OPTIMIS.

REFERENCES

[1] B. K. Alunkal, I. Veljkovic, G. V. Laszewski, and K. Amin. Reputation-based grid resource selection. In *In Workshop on Adaptive Grid Middleware*. IEEE, 2003.

[2] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu. Web Services Agreement Specification (WS-Agreement). www.ogf.org/documents/GFD.107.pdf, Visited July 2010.

[3] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the clouds: A Berkeley view of cloud computing. Technical report, Electrical Engineering and Computer Sciences, University of California at Berkeley, 2009. Technical Report No. UCB/EECS-2009-28.

[4] M. Buco, R. Chang, L. Luan, C. Ward, J. Wolf, and P. Yu. Utility computing SLA management based upon business objectives. *IBM Systems Journal*, 43(1):159–178, 2004.

[5] J.S. Chase, D.C. Anderson, P.N. Thakar, A.M. Vahdat, and R.P. Doyle. Managing energy and server resources in hosting centers. *ACM SIGOPS Operating Systems Review*, 35(5):103–116, 2001.

[6] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao. Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *In 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI'08)*, pages 337–350. ACM, 2008.

[7] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam. Managing server energy and operational costs in hosting centers. *ACM SIGMETRICS Performance Evaluation Review*, 33(1):303–314, 2005.

[8] M. Comuzzi, C. Kotsokalis, G. Spanoudakis, and R. Yahyapour. Establishing and Monitoring SLAs in Complex Service Based Systems. In *Proceedings of the 2009 IEEE International Conference on Web Services*, pages 783–790. IEEE Computer Society, 2009.

[9] Art. 29 data protection working party, work programme 2010–2011, February 2010. http://ec.europa.eu/justice_home/fsj/privacy/docs/wpdocs/2010/wp170_en.pdf, Visited July 2010.

[10] J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[11] K. Djemame, I. Gourlay, J. Padgett, G. Birkenheuer, M. Hovestadt, O. Kao, , and K. Voss. Introducing risk management into the grid. In *2nd IEEE International Conference on e-Science and Grid Computing, e-Science'06*. IEEE, 2006.

[12] K. Djemame, I. Gourlay, J. Padgett, K. Voss, and O. Kao. Risk Management in Grids. In R. Buyya and K. Bubendorfer, editors, *Market-Oriented Grid and Utility Computing*, pages 335–353. Wiley, 2009.

[13] The challenge of energy efficiency through information and communication technologies, February 2009. Parliament resolution of 4 February 2009, http://www.europarl.europa.eu/sides/getDoc.do?pubRef=-//EP//TEXT+TA+P6-TA-2009-0044+0+DOC+XML+V0//EN, visited July 2010.

[14] W. Fellows. IT is Cloud: To Infrastructure and Beyond. The 451 Group – CloudScape, June 2010.

[15] D. Gilat, A. Landau, and A. Sela. Autonomic self-optimization according to business objectives. In *1st International Conference on Autonomic Computing, ICAC 2004*, pages 206–213. IEEE, 2004.

[16] Google. Google App Engine. code.google.com/appengine, Visited June 2010.

[17] H. Grant and T. Finlayson. Cloud computing & data protection, March 2009. http://www.twobirds.com/English/News/Articles/Pages/Cloud_Computing_Data_Protection_030609.Aspx, Visited August 2010.

[18] P. Hustinx. Data protection and cloud computing under EU law. In *Third European Cyber Security Awareness Day, BSA, European Parliament*, pages 1–7, April 2010.

[19] D. Jordan and J. Evdemon (chairs). Web Services Business Process Execution Language version 2.0, 2007. http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf, Visited June 2010.

[20] A. Jøsang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43:618–644, 2007.

[21] K. Keahey and T. Freeman. Contextualization: Providing one-click virtual clusters. In *Fourth IEEE International Conference on eScience*, pages 301–308. IEEE, 2008.

[22] K. Keahey, M. Tsugawa, A. Matsunaga, and J. Fortes. Sky computing. *IEEE Internet Computing*, 13(5):43–51, 2009.

[23] J. Kephart, H. Chan, R. Das, D. Levine, G. Tesauro, F. Rawson, and C. Lefurgy. Coordinating multiple autonomic managers to achieve specified power-performance tradeoffs. In *4th International Conference on Autonomic Computing, ICAC 2007*. IEEE, 2007.

[24] J. Kephart and D. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003.

[25] J. Li, O. Wäldrich, and W. Ziegler. Towards SLA-based Software Licenses and License Management in Grid Computing. In *Proceedings of the CoreGRID Symposium*, pages 139–152. Springer Verlag, 2008.

[26] L. Liu, H. Wang, X. Liu, X. Jin, W.B. He, Q.B. Wang, and Y. Chen. GreenCloud: a new architecture for green data center. In *Proceedings of the 6th International Conference on Autonomic Computing*, pages 29–38. ACM, 2009.

[27] A. McCloskey, B. Simmons, and H. Lutfiyya. Policy-based dynamic provisioning in data centers based on slas, business rules and business objectives. In *In IEEE/IFIP Network Operations and Management Symposium (NOMS'08)*, pages 903–906. IEEE, 2008.

[28] Microsoft. Windows Azure platform. www.microsoft.com/windowsazure, Visited June 2010.

[29] Computer OECD Committee for Information and Communications Policy. Recommendation of the council on information and communication technologies and the environment (OECD guidelines) - C(2010)61, April 2010. http://webnet.oecd.org/oecdacts/Instruments/ShowInstrumentView.aspx?InstrumentID=259, Visited July 2010.

[30] P. Padala, K.G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, and K. Salem. Adaptive control of virtualized resources in utility computing environments. *ACM SIGOPS Operating Systems Review*, 41(3):289–302, 2007.

[31] Y. Poullet, J. Van Gyseghem, J. Gerard, C. Gayrel, and J. Moiny. *Cloud Computing and its Implications on Data Protection*, chapter discussion paper prepared by the Research Centre on IT and Law - CRID, page 8. Council of Europe, March 2010. www.coe.int/t/dghl/cooperation/economiccrime/cybercrime/Documents/Reports-Presentations/2079_reps_IF10_yvespoullet1b.pdf.

[32] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres, M. Ben-Yehuda, W. Emmerich, and F. Galán. The RESERVOIR Model and Architecture for Open Federated Cloud Computing. *IBM Journal of Research and Development*, 53(4), 2009.

[33] Y. Song, Y. Sun, H. Wang, and X. Song. An adaptive resource flowing scheme amongst VMs in a VM-based utility computing. In *In 7th IEEE International Conference on Computer and Information Technology (CIT 2007)*, pages 1053–1058. IEEE, 2007.

[34] J. Tordsson, R.S. Montero, R.M. Vozmediano, and I.M. Llorente. Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers, 2010. Submitted for journal publication.

[35] C. Vecchiola, X. Chu, and R. Buyya. Aneka: a software platform for .NET-based Cloud computing. In *High Speed and Large Scale Scientific Computing*, pages 267–295. IOS Press, 2009.