A Look to the Old-world[*] Sky: EU-funded Dependability Cloud Computing Research

Alysson Bessani
University of Lisbon, Faculty of Sciences, Portugal
bessani@di.fc.ul.pt

Rüdiger Kapitza
TU Braunschweig, Germany
rrkapitz@ibr.cs.tu-bs.de

Dana Petcu
Institute e-Austria Timişoara & West University of Timişoara, Romania
petcu@info.uvt.ro

Paolo Romano
INESC-ID / Instituto Superior Tecnico, Portugal
romano@inesc-id.pt

Spyridon V. Gogouvitis
National Technical University of Athens, Greece
spyrosg@mail.ntua.gr

Dimosthenis Kyriazis
National Technical University of Athens, Greece
dimos@mail.ntua.gr

Roberto G. Cascella
Inria Rennes, France
roberto.cascella@inria.fr

ABSTRACT

Cloud computing is currently the most important trend in the Information and Communication Technology (ICT) industry, and it has still not fully realized its potential. Reasons for its popularity are the opportunities to rapidly allocate vast amounts of computing resources and the fact that resources are accounted per use. While cloud computing was initiated by major industry players, academia has rapidly caught up; currently we see a vast number of cloud computing-related research efforts. However, since industry pushes development and many research aspects of cloud computing demand for large compute resources and real workloads, pure academic efforts have difficulties to address the most important issues and to have a major impact. On the other hand, academia usually tends to explore disruptive ideas that would not be addressed by industry alone.

This paper summarizes the approaches and methods of five EU-funded research projects that focus on cloud computing in general and address important issues such as security, dependability, and interoperability. These aspects have received limited attention by the industry so far. The key to success of these large joint efforts is the close collaboration between partners from academia and industry spread all over Europe. The specific projects are Cloud-TM, Contrail, mOASIC, TClouds and VISION Cloud. Besides presenting the individual projects and their key contributions, we provide a perspective on future ICT research in Europe.

[*]Although the term "old world" usually refers to Europe, Asia and Africa, we use it here only as a reference to Europe.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Cloud computing

General Terms

Reliability, Security, Design, Performance, Algorithms

Keywords

Dependability, Europe, Research projects, Clouds

1. INTRODUCTION

Cloud computing is one of the major trends in Information and Communication Technology (ICT) industry these days and still has not reached all possible markets. A major reason for its popularity is the opportunity to rapidly allocate and deallocate vast amounts of compute resources as needed. Moreover, a key aspect for the success of the cloud model is the existence of a business model (pay-per-use) that may be lucrative not only for providers but also for cloud users.

While cloud computing was initiated by major industry players, academia rapidly caught up, and currently we see a large and steadily growing number of cloud computing related research efforts. However, since industry pushes development and many research aspects of cloud computing demand for large compute resources and real workloads/field experience, pure academic works have difficulties to address the most important issues and therefore fail to have a major impact.

Recognizing the limitations of pure academic projects, the European Commission (EC) have been pushing for hybrid project consortiums composed by industry and academic parts. The idea is to foster new ideas that probably would be deemed too risky for companies alone to try and, at the same time, ensure that these ideas can find its way to innovative products and services by industry. In this paper we present an overview of five projects being funded the EC in the period of 2010-2013. All these projects deal with dependability concerns in cloud computing, considering different application scenarios (from the deployment of smart grid controllers on the cloud to the storage of massive multimedia data in media companies) and a wide range of disruptive techniques (from new programming models for Internet-scale services [36] to new dependable services based on advanced and not yet deployed techniques such as Byzantine fault tolerance [1, 31]).

The five projects discussed in this paper (Cloud-TM [2], Contrail [4], mOSAIC [8], TClouds [10], and VISION Cloud [11]) involve more than 40 partners from almost every European country, including major companies and organizations such as IBM, HP, Philips, SAP, SNIA, France Telecom, and RedHat. The total cost of these projects is around 44 million euros, with more than 75% of this volume being supported by the EC.

The objective of this paper is twofold. First, we aim to demonstrate how complementary the contributions of these five projects are and to instigate a discussion about how the results of these projects can shape European clouds in the years to come. Second, we want to disseminate some of these exciting results to a wider audience, especially in the US, since many of the activities being developed in these projects do not target publication, and thus lack a better dissemination outside Europe.

It is important to remark that these five projects are not the only ones dealing with dependability issues of cloud computing. However, they are perhaps the most prominent representatives of this exciting European research effort. For a more complete and extensive description of the cloud projects presented here (and also other cloud-related efforts), we invite the reader to consult [40].

The paper is organized as follows. Section 2 gives a brief overview of how EC funding works. Section 3 describes some of the concerns being addressed by the projects described in the paper. The following section describes the five projects. Finally, Section 5 presents some final remarks.

2. EUROPEAN RESEARCH FUNDING FOR HIGHLY INNOVATIVE ICT

Since 1984, the European Commission has been implementing funding programmes to support highly innovative research and development activities in the field of information and communication technologies. Until 2013 seven Framework Programmes (FP) have been implemented, covering eight strategic challenges, namely: Pervasive and Trusted Network and Service Infrastructures, Cognitive Systems and Robotics, Alternative Paths to Components and Systems, Technologies for Digital Content and Languages, ICT for Health, Ageing Well, Inclusion and Governance, ICT for low carbon economy, ICT for the Enterprise and Manufacturing, ICT for Learning and Access to Cultural Resources. Each challenge has specific objectives and priorities, which are being defined and detailed in the corresponding FPs. The next FP, namely "Horizon 2020" [6], will run from 2014 to 2020 covering various cooperation themes [5], such as ICT, health, transport and socio-economics, with an estimated budget of 80 billion euros.

The research and development of networks and service infrastructures was considered to be the challenge number one in the last three workprogrammes of the currently running FP7-IC. Thereby, cloud computing being an important topic that is not only key for the first challenge but for many areas of applications of ICT. In this context, multi-national and multi-disciplinary teams from academic and industrial sectors are conducting research in collaborative projects to respond to the long term challenges of the current society and the science of the future. According to [37] currently more than thirty european projects are working in the cloud computing field. Five of them are briefly summarized in this paper. Following the advent of the last calls of the workprogramme on the topic of Cloud computing, around twenty more projects are expected to start in Autumn 2012. The current trends of the research activities in Cloud computing were recently identified in [43] by the Expert Group in Cloud Computing of the European Commission and the recommendations will be taken into account in the new workprogramme.

3. CONCERNS BEING ADDRESSED

Given their size and scope, the EC-funded projects presented in this paper address many concerns relevant to their targeted use cases, however, at least five perennial concerns can be identified on these projects.

- **(The lack of) Trust on the clouds:** There is a big debate about how much one needs to trust a cloud provider. Some adopters advocate that the model itself requires complete trust in the same sense a person needs to trust a Bank[1]. Detractors point out that accidental and malicious failures that cause extended unavailability of some providers as well as information leaks and data corruption in some cloud services are more common than one would expect. In the end, there must be a middle ground in which customers would justifiably trust clouds that give extended security guarantees and/or use more than one provider for tolerating or recover from such events.
- **SLAs (Service Level Agreements):** A fundamental requirement of trustworthy cloud computing is the definition and enforcement of service level agreements between customers and providers. The wide range of applications and requirements a cloud infrastructure can host and observe, respectively, makes this problem much harder for a cloud provider than, for example, for a telecom provider. Moreover, the complexity of a cloud infrastructure and the economy of scale employed by cloud providers may require a complete reengineering of the cloud stack in order to support SLAs.
- **Portability:** One of the key impediments for moving some services to the cloud is the fear of *vendor lock-in*, when the cost of moving a service from one provider to another is so high that one needs to comply with a provider, even in adverse conditions (poor performance, higher-than-expected costs, etc.). Another im-

[1]It should be noted, however, that Banks are centuries-old organizations backed by states (i.e., if some problems happen, citizens tend to have their savings back). Unfortunately, this is not the case with cloud providers.

portant aspect of vendor lock-in is the lack of portability between clouds. This is specially true for the Platform as a Service (PaaS) model, where an application developed for a certain platform cannot easily migrate to another. In this sense, a fundamental requirement for avoiding vendor lock-in is the portability among different providers' platforms and services, which can only be achieved if the providers agree on a common platform (which is very unlikely, since the offers vary) or the services are developed with a portability layer, that adapt the specific needs of the application to a specific provider platform.

- **Programming models and infrastructures:** Cloud computing widened the horizon of distributed computing as more quickly than ever diverse distributed resources can be allocated and released. This imposes new challenges on handling system complexity. A promising approach to address such developments is to provide tailored programming models and their associated infrastructures. One prominent example is MapReduce but there will be more to come. These future programming models need to address scalability, resilience and adaptiveness to resource availability.
- **"Intelligent" storage solutions:** The sheer amount of ever-growing data demands for new solutions at all levels. While cloud infrastructures offer data storage of nearly unlimited size, the problem is to exchange large data sets and to process it. Thus, data should be moved with care and computation should be moved to the data and not vice versa.

As shown in the remaining of the paper, some of these concerns are being addressed by multiple projects. However, in most cases the approaches being taken are complementary.

4. APPROACHES UNDERTAKEN BY THE SELECTED PROJECTS

In this section we focus on five particular on-going projects (2010-2013) that we consider relevant for the subject of dependability and security in cloud computing.

4.1 Cloud-TM

The Cloud-TM project [2] aims at designing, building, and evaluating an innovative middleware platform for the development of Cloud-based services. The Cloud-TM project builds on recent results in the area of Transactional Memory (TM) [44], a programming paradigm originally proposed to simplify the development of parallel applications in multicore environments. TM integrates the familiar notion of atomic transaction as a first-class programming language construct, sparing programmers from the burden of implementing low level, lock-based synchronization mechanisms.

The Cloud-TM project aims to extend the conventional TM abstraction, traditionally confined within the boundaries of a single multi-processor machine, to seek a convergence with the distributed computing paradigm, and introduce a novel programming abstraction which combines the simplicity of TM with the scalability and failure resiliency achievable by exploiting the resource redundancy proper of large scale cloud infrastructures [42].

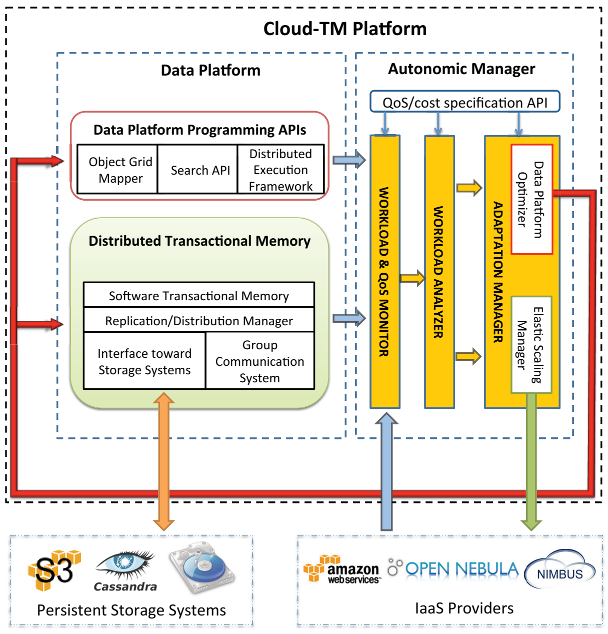
Cloud-TM Platform
Data Platform
Autonomic Manager
QoS/cost specification API
Data Platform Programming APIs
Object Grid Mapper
Search API
Distributed Execution Framework
Distributed Transactional Memory
Software Transactional Memory
Replication/Distribution Manager
Interface toward Storage Systems
Group Communication System
WORKLOAD & QoS MONITOR
WORKLOAD ANALYZER
ADAPTATION MANAGER
Data Platform Optimizer
Elastic Scaling Manager
S3
Cassandra
Persistent Storage Systems
amazon webservices
OPEN NEBULA
NIMBUS
IaaS Providers

Figure 1: Overview of Cloud-TM architecture.

4.1.1 *Architectural Overview*

Figure 1 provides an architectural overview of the Cloud-TM platform. The Cloud-TM platform is formed by two main components: the Data Platform and the Autonomic Manager.

The Data Platform is responsible for storing, retrieving and manipulating data across a dynamic set of distributed nodes, elastically acquired from the underlying IaaS Cloud provider(s). It exposes a set of APIs, denoted as "Data Platform Programming APIs" in Figure 1, aimed at increasing the productivity of Cloud programmers from a twofold perspective:

1. Storing and querying data into/from the Data Platform using the familiar and convenient abstractions provided by the object-oriented paradigm, such as inheritance, polymorphism, associations.
2. Taking full advantage of the processing power of the Cloud-TM platform via a set of abstractions that will hide the complexity associated with parallel/distributed programming, such as load balancing, thread synchronization and fault-tolerance.

The backbone of the Data Platform is represented by a highly scalable, elastic and dynamically reconfigurable Distributed Software Transactional Memory (DSTM). In order to maximize the visibility and impact of the project's results, Cloud-TM selected as its reference DSTM platform one of the mainstream open-source in-memory transactional platforms, namely Red Hat's Infinispan. Infinispan is a recent in-memory transactional data grid [34] designed from the ground up to be extremely scalable, and is being enhanced during the project with innovative algorithms for data replication and on-line self-tuning.

The Autonomic Manager is the component in charge of automating the elastic scaling of the Data Platform, and of orchestrating self-optimizing strategies aimed to dynamically reconfigure the data distribution and replication mechanisms. Its topmost layer exposes an API allowing the specification and negotiation of QoS requirements and budget constraints. The Autonomic Manager leverages on pervasive monitoring mechanisms that do not only track the utilization of heterogeneous system-level resources (such as CPU, memory, network and disk), but characterize also the workload sustained by the various subcomponents of the Data Platform and their efficiency. The stream of raw data gathered by the Workload and Performance Monitor is then filtered and aggregated by the Workload Analyzer, which generates workload profiling information and alert signals for the Adaptation Manger. Finally, the Adaptation Manager hosts a set of optimizers relying on techniques of different nature, including analytical, simulative and machine-learning, to self-tune the platform's components and control its auto-scaling with the ultimate goal of meeting QoS/cost constraints.

4.1.2 Selected results in the dependability area

In the following we overview some recent results achieved by the Cloud-TM project in the area of dependable data management. In particular we review three results addressing different aspects of transactional data replication, namely high scalability, autoscaling and self-tuning.

Scalable transactional replication. A common trait characterizing the new generation of cloud data platforms is the adoption of a range of weak consistency models, such as eventual consistency [24], restricted transactional semantics (e.g. single object transactions [33], or static transactions [13]), and non-serializable isolation levels [15]. By embracing weak consistency, these platforms achieve unprecedented scalability levels. On the down side of the coin, weak consistency models expose additional complexity to application developers, forcing them to deal with the idiosyncrasies caused by concurrency, network partitioning and/or failures. Therefore, a crucial research question is whether consistency and scalability are actually two mutual exclusive qualities, or whether, there exist any sweet spot in the trade-off between scalability and consistency that allows to design highly scalable data replication protocols while exposing simple and intuitive consistency semantics.

In Cloud-TM we have addressed this fundamental issue by designing and integrating in our platform GMU [36], a genuine partial replication protocol for transactional systems, which exploits an innovative, highly scalable, distributed multiversioning scheme. GMU never blocks or aborts read-only transactions and spares them from distributed validation schemes. Unlike existing multiversion-based solutions, GMU does not rely on a global logical clock, hence avoiding global contention points that would limit system scalability. GMU has been evaluated in large scale public and private cloud platforms of up to a hundred nodes with challenging OLTP workloads such as the industry standard TPC-C benchmark, showing linear scalability and negligible overheads with respect to much weaker isolation levels such as SQL's repeatable read (see Figure 2).

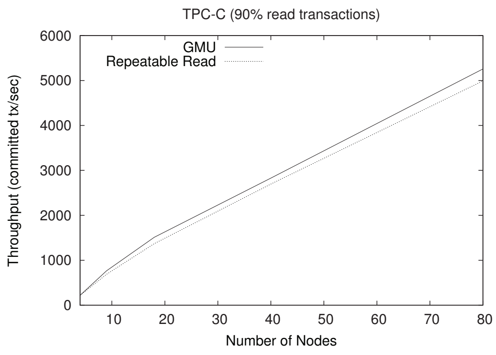

Figure 2: Performance of the GMU protocol [36].

GMU guarantees the Extended Update Serializability (EUS) isolation level, a consistency criterion that is sufficiently strong to ensure correctness even for very demanding applications (such as TPC-C), but also weak enough to allow efficient and scalable implementations, such as GMU. Further, unlike several relaxed consistency models proposed in literature, EUS has simple and intuitive semantics, thus being an attractive, scalable consistency model for ordinary programmers

Auto-scaling. Existing cloud platforms allow non-expert users to provision a cluster of any size on the cloud within minutes. However, removing the system administrator and the traditional capacity-planning process from the loop shifts the non-trivial responsibility of determining a good cluster configuration to the non-expert user. Unfortunately, forecasting the performance of data centric applications while varying the scale of the underlying platform is extremely challenging. In fact, the performance of distributed data platforms tend to exhibit strong non-linear behaviors as the number of nodes in the system grows, as a consequence of the simultaneous, and often inter-dependent, effects of contention affecting both physical (computational, memory, network) and logical (conflicting data accesses by concurrent transactions) resources [26]. Due to these complexities, auto-scaling mechanisms currently offered by commercial cloud support only simple reactive provisioning policies based on user defined thresholds on resource (e.g., CPU or memory) utilization. However, no guarantee is provided on the impact of the auto-scaling policies on key application level performance indicators, such as throughput or response time, which are essential for the definition of any Service Level Agreement.

The Cloud-TM project has attempted to fill this gap by introducing TAS (Transactional Auto Scaler) [26], an elastic-scaling system (see Figure 3) that relies on a novel hybrid analytical/machine-learning-based forecasting methodology to predict the performance achievable by transactional in-memory data stores in face of changes of their scale [26]. Applications of TAS range from on-line self-optimization of in-production applications, to the automatic generation of QoS/cost driven elastic scaling policies, and what-if analysis on the scalability of transactional applications.

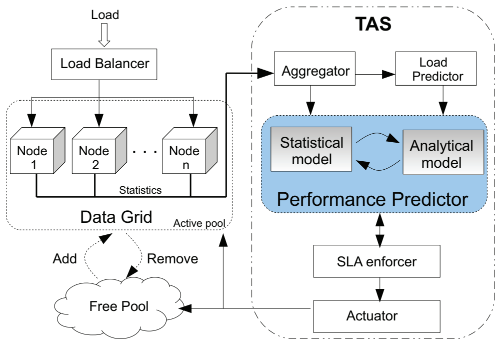

Figure 3: Architecture of the Transactional Auto Scaler [26].

Self-tuning replication. Decades of literature and field experience in the area of data replication have brought to the development of a plethora of approaches for state consistency in distributed platforms, and taught a fundamental, general lesson: no universal, one-size-fits-all solution exists that can achieve optimal efficiency across all possible kinds of workloads and for any level of scale of the system. This issue is hence particularly exacerbated in Cloud computing platforms due to the feature that is regarded as one of the key advantages of the cloud: its ability to elastically acquire or release resources, dynamically varying the scale of the platform in real-time to meet the demands of varying workloads. This means that in order to maximize efficiency (i.e. minimizing operational costs, in the pay-per-use pricing model) data management middleware needs to adapt its consistency mechanisms in order to ensure optimal performance for every workload and at any scale.

An important step achieved by the Cloud-TM project towards the fulfillment of this goal is PolyCert [23]. Polycert is a polymorphic data replication protocol that allows for the concurrent co-existence of different state of the art transactional replication protocols, relying on machine-learning techniques to determine the optimal certification scheme on a per transaction basis. By self-tuning the replication strategy on the basis of current workload, PolyCert can achieve a performance extremely close to that of an optimal non-adaptive protocol in presence of non heterogeneous workloads, and significantly outperform any non-adaptive protocol when used with complex applications generating heterogeneous workloads.

4.1.3 *Summary*

Cloud-TM aims at defining a novel programming paradigm to facilitate the development and administration of cloud applications. To this end, the Cloud-TM project is developing a self-optimizing Distributed Transactional Memory middleware that will expose a familiar object-oriented programming paradigm and spare programmers from the burden of coding for distribution, persistence and fault tolerance, letting them focus on delivering differentiating business value.

In order to achieve optimal efficiency with any workload and at any scale, the Cloud-TM middleware will integrate pervasive autonomic mechanisms that will serve the twofold purpose of automating resource provisioning and self-tuning the various layers of the platform.

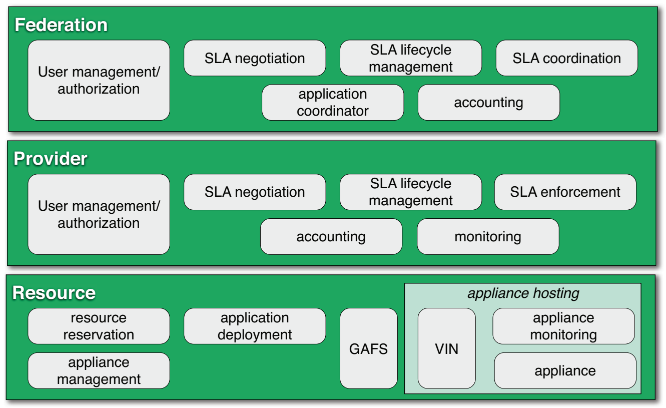

Figure 4: Overview of Contrail architecture.

4.2 Contrail

Contrail [4] is an open source integrated approach to virtualization, which aims at offering Infrastructure as a Service (IaaS) services, services for federating IaaS clouds, and Contrail Platform as a Service (ConPaaS) services on top of federated clouds. In Contrail, the user is relieved from managing the access to individual cloud providers and can focus on specifying the service or application. The application will be then deployed on few clouds selected from a large set of heterogeneous cloud providers. These providers can implement a different cloud technology, have different hardware, or offer different types of guarantees.

Contrail implements a *dependable* cloud by guaranteeing the availability of the computational resources and having strict guarantees in terms of quality of service (QoS) and quality of protection (QoP), that customers can specify in the Service Level Agreement (SLA), when submitting their requests, and monitor during the execution of the application. The Contrail Federation is the primary access for the customers. It is the entity entitled to select the best provider for serving the customers' requests, and to negotiate and enforce the SLAs even on unreliable providers. Thus, a customer of Contrail only needs to submit the distributed application, along with its runtime configuration, and specify the requirements in an OVF [12] and SLA documents respectively. Then, the Federation ensures that the providers' resources are utilized as needed for offering an elastic, dependable, and trustworthy cloud service. These qualities enable customers to rely on cloud computing as an external source for their data management and as processing facilities for creating their business on top.

Figure 4 depicts the Contrail architecture. It is designed to be extensible, allowing the reuse of some components in different layers, and to give the possibility to exploit components independently. As such it is organized in modular components separated by well-defined interfaces and structured in layers to specifically address, from the top to down, the federation, the provider, and the resources. The federation layer is the entry-point for users, who register and authenticate to use the Contrail services; users interact with this layer to negotiate SLAs, to submit and monitor their applications. The federation layer is then in charge of interacting with the different cloud providers, enabling seamless access to their resources. The Contrail Federation [21] implements federated identity management, for authentication and authorization, and integrates security mechanisms to provide strict security guarantees expressed as quality of protection (QoP) terms. The SLA terms for security (QoP) and performance guarantees (QoS) are used to select the most suitable cloud providers to deploy the user's application based on the resources available and the providers' *reputation*, matching the level of performance and trustworthiness required by the application. The Federation then proceeds to negotiate proper SLA terms with each provider in a transparent way for the users. In this phase, a high degree of interoperability could be achieved thanks to the Virtual Execution Platform (VEP) [28, 29] service (see Figure 5) enabling the federation to manage the resources of public and private cloud providers regardless of the hardware and the technology implemented.

The provider layer implements the business part of a cloud provider since it negotiates and enforces SLAs, monitors the application and does the accounting of the resources. This layer is the sole interacting with the Contrail Federation. The resource layer manages the physical resources of a cloud provider. In Contrail, each cloud provider runs a copy of the VEP software which in turn seamlessly integrates its resources with the Contrail Federation. The separation of the provider and resource layers have a two fold meaning: a cloud provider can have many data centers, and the cloud provider could use the management services of the provider layer to run their business independently or in addition to the Contrail Federation.

A key objective of the Contrail project is to provide a reliable cloud platform. The gateway is the Contrail Federation and three other important components are VEP, GAFS, and VIN. VEP [28] is a reliable application deployment platform that is resilient to operational failures and which supports secure management of user data with a strong guarantee for QoS. It is an open source technology implementing standards and offers the deployment of end-user applications independently from the underlying platform, providing the needed support for interoperability to the Contrail Federation. GAFS (Global Autonomous File System) is a reliable and highly available storage service implemented with XtreemFS [30]. It is used both to store VM images and system logs, and as a scalable Storage as a Service for Cloud users and applications. GAFS provides scalability and elasticity, and implements security mechanisms for data access control and encryption for data in transit. It also allows to specify the level of protection of the stored data and the location of the storage due to specific legal requirements or negotiated QoS terms, e.g., low latency and high throughput. VIN (Virtual Infrastructure Network) [20] is responsible for managing all communication within a Contrail application and maintaining a stable network when resources are added or removed to an elastic application. It creates a dedicated private network per application, which is deployed in an isolated environment and can provide different security levels for the communication.

Another key objective of Contrail is to provide elastic PaaS services on top of federated clouds. This is achieved thanks to ConPaaS (the Contrail PaaS) [41] component, which will directly interact with the Federation to use services and features such as user management, SLA handling, and application deployment. The tight integration with the Federation will ensure that a ConPaaS service can be deployed over different cloud providers to guarantee elasticity within the constraint of the negotiated SLA, thus integrating security, availability and performance guarantees for a reliable execution. As a standalone component, ConPaaS [41] already ensures that services can deploy themselves on the cloud, be self-managed, elastic, and scalable. A ConPaaS service can monitor its own performance and increase or decrease its processing capacity by dynamically (de-)provisioning instances of itself in the cloud.

4.2.1 Contrail challenges for a dependable cloud

This section overviews how Contrail tackles major challenges for a dependable cloud.

Security and trustworthiness. Security and trustworthiness of the cloud infrastructure is one of the major concerns for customers willing to use cloud computing for running their applications and storing their data. Proper security can only be achieved by adequately protecting them against unauthorized access or modification. Contrail integrates authentication and authorization mechanisms at the federation layer having the Federation in charge of authenticating the users and handling user identities and credentials to act on behalf of the users seamlessly [21]. In Contrail, the entities operating the federation and the cloud infrastructure represent independent security domains. As such a user federation identity is used to match the credentials managed at the federation layer with those at the cloud providers deploying the application. The Contrail authorization system is meant to protect a user application and data from unauthorized access of others, and contextually the cloud provider from a misuse of the resources. The authorization is based on the concept of mutual attributes, associated with the resources or users, and continuous control to evaluate whether possible modifications of attributes may lead to a failure in satisfying previously defined security policies. Contrail also provides an *isolated* environment for each application via a dedicated secured virtual network (VIN in Figure 4) to meet privacy requirements and avoid other users or even the cloud provider to interfere with the application itself. The level of privacy for an application and potential legal requirements, such as geographical location of data storage or computing resources, are specified as Quality of Protection (QoP) terms in the SLA.

Availability. The availability of the resources in cloud computing is of utmost importance both for satisfying the request of the users and to implement elasticity of cloud services, which requires cloud providers to allocate on-demand resources. The Contrail Federation integrates heterogeneous public and private clouds, and gives users the possibility to deploy an application over multiple clouds seamlessly. For instance in case of a request for additional resources, which are not available at the cloud provider hosting the application, the Federation can take over and locate resources in an alternative provider to meet the customers' requests. The portability of an application or part of it on a different cloud provider could be an issue because of the technology making difficult the interoperability across federated clouds. In Contrail, interoperability is supported via the VEP service which is able to manage the deployment of an application regardless of the underlying technology and it will rely on the support of standards to describe distributed cloud applications (e.g., OVF [12]) and open protocols to build and manage them (e.g., OCCI [9]).

Sharing cloud resources with other users exposes the customers and the cloud provider to possible misuse of the resources, which might not be anymore available in case of outages or network misconfiguration. Contrail addresses these issues by ensuring proper isolation of the applications and control over the access to the resources. Moreover, the monitoring infrastructure proposed in Contrail collects information about resource usage. The monitoring data is used to take actions both at the provider and federation layer to manage the available resources efficiently during the execution of the application or to do advanced reservation before deploying an application.

Reliability and performance guarantee. On the one hand a cloud provider should contractually reach the application performance objective to ensure a reliable application execution. On the other end, the provision of new resources to meet this objective should be done at the minimum cost for the customer. Contrail uses SLAs to negotiate QoS terms; the Federation is then in charge of locating the best cloud provider and monitoring the applications. To provide resilience to failures and prevent a potential violation of SLAs or implement fault tolerance, Contrail supports application snapshot via the VEP service. The snapshot is a single OVF document describing all active Virtual Machines (VMs) with links to their respective saved disk images. The Federation can then proactively decide to migrate the whole or part of the application to another provider if more resources are needed to satisfy the QoS or a failure has been detected. In such a scenario, the Contrail software is able to satisfy the user needs for the deployment of elastic and scalable applications guaranteeing performance and fault tolerance.

4.2.2 *Application deployment and SLA management*

Figure 5 depicts the deployment of an application in Contrail. A user of the Contrail system authenticates with the Federation portal and negotiates the deployment of the application, described in the OVF [12] standard. This format is portable, being platform neutral, and is extensible by the users if needed. The user negotiates with the Federation the Service Level Agreement (SLA), which indicates the performance objectives (QoS) and level of security requested (QoP). In Contrail, there are three types of SLA templates: *specific* to an OVF descriptor of an application or service (in Figure 5), *generic* for multiple applications of the same user, or *hybrid* as it is an intersection of the previous two and refers to a specific section of the OVF descriptor. Upon verifying the available resources and the requested QoS and QoP, the Federation can decide to deploy an application on a single cloud provider or it can split it across multiple

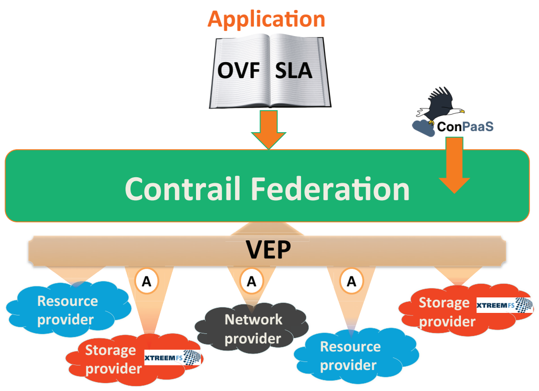

Figure 5: Application deployment in Contrail.

providers (see Figure 5). Elasticity rules are also managed via SLAs, and other cloud providers can be selected at runtime when extra resources are requested.

Each Contrail cloud provider has a SLA management component (SLA negotiation and SLA enforcement in Figure 4) and it can negotiate with the Federation long term SLAs, such that the Federation has the means to manage a user request without negotiating new SLAs. Negotiated SLAs associated to an application are communicated to the VEP component in order to be applied. The VEP is in charge of provisioning resources from the resource layer in conformance with negotiated SLAs, of monitoring the application during the whole lifecycle and of providing monitoring data to the SLA enforcement component. The monitoring data are used to detect a potential violation of SLAs. This is done first at the provider layer, where the SLA enforcement component reacts through requests to the VEP if more resources are available, otherwise at the federation layer, where required actions will be taken to satisfy the negotiated SLA. If an application is split across multiple cloud providers, each cloud provider monitors and enforces the SLA for the part of application it deploys and the Federation verifies on the aggregated monitoring data that the SLA negotiated with the user is not violated. Contrail also offers PaaS services with ConPaaS (see Figure 5). The integration of ConPaaS in the Contrail software stack [3] will guarantee the deployment of PaaS services by exploiting all the Federation features, such as monitoring and SLA enforcement.

4.2.3 *Summary of Contrail dependability features*

The main results of Contrail that provide solutions for dependable cloud services are the following:

- A Federation service integrating security to select the best matching resources (with variable prices) offered by multiple cloud providers;
- SLA negotiation and enforcement via monitoring capabilities to guarantee QoS and QoP terms;
- VEP to implement interoperability for deploying applications in a dynamic and heterogeneous environment;
- ConPaaS to offer an open-source runtime environment for easy and scalable applications in the cloud.

Addressing these important challenges is fundamental to support large user communities formed of individual citizens and/or organizations relying on Cloud resources for their business applications.

4.3 mOSAIC

The mOSAIC Consortium [8] has promise to deliver an open-source API and a PaaS which allows to port Cloud-enabled applications from one Cloud infrastructure service provider to another, and which allows to select the Cloud service at the deployment time.

4.3.1 Approach

The first step to keep the mOSAIC's promise was the design of a set of open APIs that: (a) introduces new level of abstractions in order to reach the promise of Cloud computing to make the infrastructure programmable; (b) uses an event-driven approach and message passing as being most appropriate for Cloud computing paradigm (c) is instantiated by proof-of-concept implementations in popular programming languages, like Java and Python.

While other open APIs (like jClouds, libcloud, SimpleCloud, OpenStack, etc.) are trying to provide an uniform API that abstracts the functionality of a certain type of the Cloud resource (virtual machine, storage, networking) by keeping the programming style close to the one of the native API of the Cloud resource, the mOSAIC's APIs try to improve the efficiency on the application side by removing restrictions like the requirement for synchronous communications or REST interfaces.

The new level of abstractions are represented by: (a) Drivers equivalent with the above mentioned APIs, close to the native APIs (any of the mentioned things can be a Driver); (b) Connectors that are representing the operations with a certain type of Cloud resource (e.g., message queues, key value store, etc.) in a specific language and independent from the programming style of Drivers; (c) Cloudlets that are expressing the reactions of the applications to the events related to the Cloud resources allocated to the applications; (d) Interoperability layer that acts as proxy between Connectors and Drivers. Details of these concepts as well the results of preliminary tests showing the efficiency of such approach when compared with classical synchronous and REST based architecture can be found in [39].

Beyond using open APIs, the portability can be sustained by open protocols, open standards, semantics, and abstraction layers like reference architectures or mediator services. mOSAIC is using OCCI as open protocol in its brokerage mechanisms and CDMI as open standard at Driver level, as well as a complex semantic engine that help the application designer to find the right functionality or the right Cloud service (described in terms of a particular Cloud ontology).

The second step to keep the mOSAIC's promise was the design of a PaaS that: (a) is open-source and deployable; (b) helps the application developers to deploy the application in one or multiple Clouds without the need to set the execution environment; (c) allows the development of the application on desktop after which its porting to personal, private or public Cloud can be done smoothly; (d) monitors the execution of the application allowing its elasticity at the level of components; (e) can deal with long-running applications like web applications, as well with data or computational intensive scientific applications.

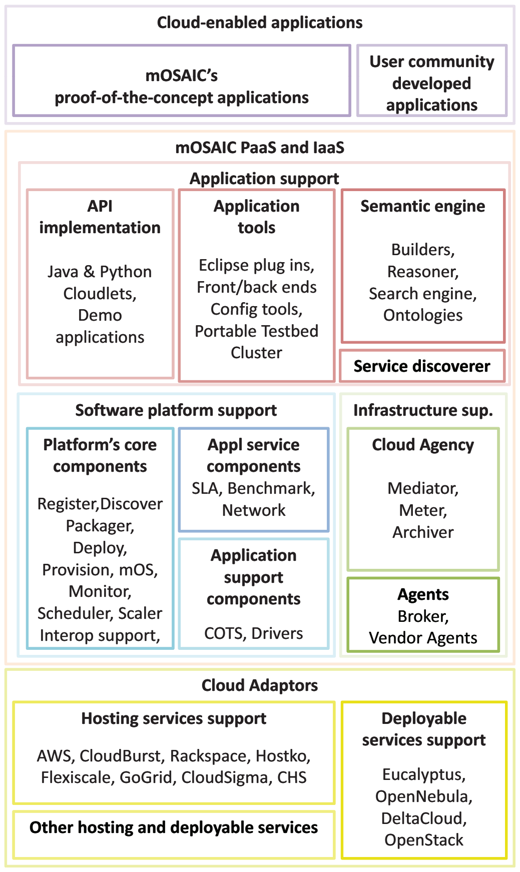
Cloud-enabled applications
mOSAIC's proof-of-the-concept applications
User community developed applications
mOSAIC PaaS and IaaS
Application support
API implementation
Java & Python Cloudlets, Demo applications
Application tools
Eclipse plug ins, Front/back ends Config tools, Portable Testbed Cluster
Semantic engine
Builders, Reasoner, Search engine, Ontologies
Service discoverer
Software platform support
Platform's core components
Register,Discover Packager, Deploy, Provision, mOS, Monitor, Scheduler, Scaler Interop support,
Appl service components
SLA, Benchmark, Network
Application support components
COTS, Drivers
Infrastructure sup.
Cloud Agency
Mediator, Meter, Archiver
Agents
Broker, Vendor Agents
Cloud Adaptors
Hosting services support
AWS, CloudBurst, Rackspace, Hostko, Flexiscale, GoGrid, CloudSigma, CHS
Other hosting and deployable services
Deployable services support
Eucalyptus, OpenNebula, DeltaCloud, OpenStack

Figure 6: General overview of mOSAIC's PaaS architecture.

4.3.2 Architectural Overview

A general overview of the mOSAIC's PaaS is provided in Figure 6. The implementation in Java of the API set and the open-source part of the PaaS are available at [7].

The targeted applications are the ones built from components that are able to be multiplied or reduced in the execution phase in order to keep the application's key performance indicators between the desired thresholds. Simple proof-of-concept web applications are provided with the code (one being exposed in [38]), while the complex applications that have motivated the developments and which are providing currently the first feedback are partially described in [25].

An application descriptor exposing the application's components and communications between them is interpreted by the platform and an adequate request for e-infrastructure services is forwarded at deployment stage to a Cloud agency (details in [45]) that, after the service level agreement negotiations, is entitled to return the links to the allocated Cloud resources.

Various proof-of-the-concept adaptors to Cloud services are currently in development, half of them for public Clouds offered by European providers or for open-source deployable software allowing to build Private or Hybrid Clouds. The repository of re-usable components includes, beyond the Connectors, Cloudlets and Drivers, open source deployable services like Riak for key-value store or RabbitMQ for message passing (depicted by the acronym COTS in Figure 6).

4.3.3 Results in the dependability area

From the point of view of dependable Cloud computing, mOSAIC deals with the followings topics: fault tolerance, availability, reliability, safety, and maintainability.

To ensure the *fault tolerance*, several decisions were taken at the design of the new set of APIs. The application components are requested to communicate only via message queues that are seen as Cloud resources. In the case of faults in one component instance the messages are redirected to another instance of the same component. Several instances of the same component are managed by a Container that is entitled to implement the fault tolerance mechanisms.

To ensure the *availability* of the application on long term or in the case of breakdowns of the underlying Cloud services, the application deployer can request to the Cloud agency the re-allocation of new Cloud resources. Furthermore, the application can be developed and partially maintained also on the developer desktop (through the usage of the Portable Testbed Cluster).

To ensure the *reliability* of the application, the number of Containers that are managing a certain number of instances of the application components can be specified at the deployment phase. Different Containers will be scheduled usually in different virtual machines.

In what concerns the *safety* of the developed and deployed applications, it is necessary to underline the advantages of the event-driven approach. Event-based programs tend to use a single thread to process events, avoiding concurrency problems. Moreover, the event-based programs running under heavy load are more stable than the threaded programs.

To ensure the *maintainability*, the components of the application can be stopped and restarted during the application execution. This is possible due to discovery services that are part of the platform, as well as due to the asynchronous communications and message queuing system.

4.4 TClouds

The TClouds project [10] aims to improve the security and robustness of cloud services. As of today, cloud providers offer only a few, if any, explicit security guarantees in their SLAs. The providers are generally confronted with a demand for more robust services from their customers. This is underlined by a number of security incidents and failures of cloud services, which have received public attention, sometimes deserved and sometimes not.

TClouds explores methods from security and dependability in operating systems, virtualization, and distributed systems for increasing the resiliency and security of cloud infrastructures and platforms. The project centers around a reference architecture [46] that rests on the following three observations:

1. A cloud scenario has dependability and security needs that cannot be met by the application layer alone, requiring security-specific solutions to be provided at lower layers of the cloud architecture (infrastructure and platform).

2. Proprietary approaches to achieving resilience in a cloud infrastructure and platform make it more difficult and more expensive to migrate and interoperate between providers, creating vendor lock-in and excluding competition. Open, standards-based mechanisms are required instead.

3. Wherever there is a single point of failure, be it at a technical or an organizational level, any high-resilience objective may be compromised. This holds also for multiple federated clouds that share common or related management and trust domains. This requires a genuinely diverse multi-provider approach.

These observations lead to the development of a set of solutions representing different instantiations of the reference architecture. These solutions comprise mainly the construction of either a trusted cloud infrastructure and the design of dependable cloud services using the available public cloud offers.

4.4.1 Selected project results

In the following we discuss the main research avenues pursued by TClouds.

Legal requirements. The TClouds project acknowledges that many of the impediments to adopting cloud computing stem from legal uncertainty and from issues with business models. From the legislation point of view, the project is investigating legal challenges associated with the flexibility of cloud computing. The reality of multinational cloud platforms raises questions regarding which legislations govern the handling of personal identifiable information (PII). The data protection legislation of the EU is applicable whenever the cloud provider (or a unit thereof) is located or uses equipment for data processing in a member state of the EU. Therefore, any server farm, data center, or specific geographic component in the EU results in applicability. The EU data protection framework restricts the transfer of PII to outside the EU. It demands adequacy of data protection standards of the recipient to guarantee sufficient safeguards of the individual's right to information self-determination. Only few countries meet these high expectations of adequate privacy provisions since 75% of states worldwide do not have comprehensive data protection laws, including the USA. Transferring PII to these states is only allowed under very strict contractual conditions (e.g., if the information cannot be related with some citizen). The European Commission established a self-certification framework for US companies, the US-EU Safe Harbor framework. This allows

US companies to self-certify adherence to privacy principles that should provide adequate data protection standards within this company.

Verifying virtual infrastructures. A second contribution of TClouds is a set of tools to assess the security of services deployed on public cloud platforms. One of these tools was used in a recent study for exposing that many of the OS images found in the Amazon Cloud App Store have backdoors and other vulnerabilities, which renders any client running such an image prone to simple attacks [18]. This tool, called AmazonIA (Amazon Image Attack), allows one to scan an OS image for backdoors before using it and thereby increases the confidence in the proper security configuration of the image.

Another tool devised in the project aims to verify the information flow in of heterogeneous virtualized computing infrastructures [17]. The tool inspects VM configuration files, builds a graph model, and computes the potential information flow between the components, in order to identify possible leaks, attacks, and security-policy violations.

Trusted cloud infrastructures. One of the key objectives of TClouds is to build secure private clouds in such a way that they may run safety-critical and sensitive services, which cannot be done in a public cloud today. This objective is being addressed through the exploitation of trusted computing (TC) technology. The idea is to build data centers with TC-enabled servers (e.g., that possess a TPM chip or another hardware root-of-trust) that can provide such advanced security properties. In particular, TClouds members are extending the OpenStack resource scheduler with remote attestation capabilities to ensure that only certified VMs are deployed in a given infrastructure.

An orthogonal but related solution being devised in the project is a system for the management of trusted virtual domains (TVDs) composed of sets of interconnected virtual machines [19]. The VMs of a TVD are interconnected in such a way that their data transfers never leave the TVD. Remote communication between compartments over untrusted networking infrastructure is secured by a security kernel deployed on each physical machine. This kernel is deemed secure since its code is verified through remote attestation using TC technology.

Every information flow within a TVD or between TVDs is also controlled by the security kernel and is only granted if it complies with the security policy. The security policy is enforced by a trusted organization manager component (TOM) that manages all TVDs and appliances in the organization through a trusted management channel between itself and the security kernel of each machine.

Resilient cloud services. Most cloud infrastructures and platforms rely on a set of critical services in order to operate correctly. Examples of such services are coordination, authentication, logging, metadata store, databases and so on. Many of these services employ state-machine replication protocols for high-availability in face of crashes and asynchrony. TClouds builds on the recent development of practical Byzantine fault-tolerance (BFT) systems in order to create cloud-management services that can effectively tolerate adversarial influence beyond crashes, such as value-domain errors, malicious attacks, or Byzantine faults. The project addresses this topic by designing and implementing a robust BFT replication library that can be used for replicating any (deterministic) service. This library is called BFT-SMaRt, and is freely available on the web [1]. Contrary to previous works on the field, the focus of BFT-SMaRt is not on providing new protocols and algorithms, but instead to make a subset of state-of-the-art techniques work on a stable system.

By exploiting the trusted computing technology available in TClouds-enabled private clouds, the project also improves the replication protocols themselves. In particular, TClouds has developed two resource-efficient BFT replication protocols that tolerate up to f faulty replicas: one uses $2f + 1$ replicas that may be hosted in diverse data centers [47] and the other uses only $f + 1$ replicas residing in the same datacenter [31]. Prototypes for these replication protocols are being implemented using the BFT-SMaRt codebase.

Programming models & platforms. Another objective of TClouds is to improve the resilience of widely used cloud platforms and programming models. A relevant result in this direction was a MapReduce platform capable of tolerating (non-malicious) Byzantine faults that uses several techniques to avoid incurring on prohibitive costs [22].

Cloud-of-clouds services. One last strand of work in TClouds addresses the combination of multiple, independent cloud services into one more resilient "cloud-of-clouds" service. The underlying idea is the same as for building resilient single-domain cloud services from replicated components: If one whole cloud represents a unit of failure, then a resilient federation of multiple cloud services should result in a service with better availability and more security. This idea is pioneered by TClouds by combining resources from a diverse set of commodity public clouds.

Through using BFT replication mechanisms, this approach is able to tolerate faults and intrusions on up to a certain number of providers. However, the BFT mechanisms must be adapted to the different inter-cloud environment from the intra-cloud setting. Several challenges need to be addressed: replication mechanisms are required that (1) can tolerate a large spectrum of faults and security threats, (2) can cope with Internet communication unpredictability to effective make multi-cloud systems run smoothly, (3) are compatible with most public clouds available without changing their infrastructures or requiring their cooperation, and (4) minimize the replication resource overhead to make the solution economically viable for practical applications. These challenges cannot be completely addressed by the current replication protocols and thus new techniques must be devised.

The first cloud-of-clouds service developed at TClouds is the *DepSky object storage* system [16], which uses object storage services from diverse cloud providers (e.g., Amazon S3, Rackspace Files) to build a virtual trusted object storage. In a nutshell, the DepSky protocols address the requirements mentioned before in the following way. First, DepSky tolerates arbitrary (a.k.a. Byzantine) faults in order to cope with all possible behavior of a fraction of providers. Second, the protocols are completely client-based, in the sense that no specific code needs to be deployed in the clouds. DepSky assumes that the clouds provide storage service with standard operations for managing objects and containers (put, get, list, etc.). Finally, DepSky employs both secret sharing and erasure codes to ensure confidentiality and space efficiency for the stored data, respectively. This solves two important problems: (1) no single cloud has access to the information stored in itself and, at the same time, there is no need for key distribution protocols; (2) storing data in N clouds cost less than N times as much as single cloud storage since only a portion of the data is stored in each cloud.

The DepSky system deals with the problem of data replication using a cloud-of-clouds. A more complex problem that is currently being addressed is how to use the computing resources available on public clouds to build dependable cloud-of-cloud computing services. The technical challenges of providing a cloud-of-clouds computing platform are substantially harder than for the storage case, since the required replication protocols for general services (e.g., state machine replication) usually do not work very well on WANs. The initial approach to cope with this problem is to extend BFT-SMaRt to achieve better performance in WANs, and then build cloud-of-cloud services using it.

4.4.2 Summary

TClouds tackled the problem of dependable and secure Cloud computing considering a general reference architcture that can be instantiated in different ways, as reflected by the broad scope of the project results. This approach takes in consideration the fact that different solutions are required at different levels, depending on the applications requirements and constraints, and that there is no silver bullet for Cloud dependability. Accordingly, the project provides only a set of tools and methods that need to be adapted for specific application scenarios.

4.5 VISION Cloud

The goal of the VISION Cloud project [11] is to "*provide reliable and effective delivery of data-intensive storage services, facilitating the convergence of ICT, media and telecommunications*". To this end five areas of innovation have been identified and are driving the VISION Cloud platform: (1) content is managed through data objects associated with a rich metadata model, (2) avoidance of data lock-in by enabling the migration of data across administrative domains, (3) avoiding costly data transfers by moving computations close to the data through programming agents called *storlets*, (4) enabling efficient retrieval of objects based on their content, properties and the relationships among them and (5) providing strong QoS guarantees, security and compliance with international regulations. More details on the project aims and architecture can be found in [32].

4.5.1 Data and Account Model

In VISION Cloud the unit of storage is a data object. A data object contains data of arbitrary type and size, has a unique identifier that can be used to access it and has metadata associated with it. Each data object resides within the context of a single container. Containers are the unit of placement, reducing not only the frequency of making global placement decisions, but also the size of the location information that has to be stored globally. Metadata can be associated with both data objects as well as containers. We distinguish between two categories of metadata, namely user and system metadata. While the semantics of the former are transparent to VISION Cloud, the platform provides the facilities needed to create, update and make queries on it. System metadata, on the other hand, has concrete meaning to the Cloud storage system. It either directs the system how to deal with the object (e.g., access control, reliability, performance requirements, etc.), or provides system information about the object (e.g., size, creation time, last access time, etc.) to the user.

The account model of VISION Cloud consists of tenants, sub-tenants and users. A tenant is an organization that subscribes to the platform's services. A tenant may represent a commercial firm, a governmental organization, or any other organization, including any group of one or more individual persons. It is the entity that negotiates Service Level Agreements (SLAs) with VISION Cloud and is billed accordingly. A tenant may also define subtenants. Subtenants can be viewed as different departments of the same organization. A firm consisting of an R&D Department and HR Department could therefore constitute different subtenants of the same tenant. This allows for different service levels to be set according to the requirements of each department, while also providing for isolation where needed. Thus, a commercial firm is able to "map" its business structure directly on the underlying storage. A user is the entity that actually consumes the storage services provided. For auditing, billing and security reasons [14, 27], every operation needs to be associated with a user.

4.5.2 Approaches addressing dependability

SLA Management, in VISION Cloud, has a central role, as a lot of management decisions come from the SLA under which a container is created. SLAs in VISION Cloud are dynamic, in the sense that, contrary to current commercial offerings, a tenant is able to negotiate its terms, cost and penalties with the platform [35]. The tenant is thus able to define various requirements to the platform, such as:

- Latency and throughput;
- Durability levels, by which the probability of data loss is defined;
- Availability, by which the probability that the data is available when requested is defined;
- Geographic preference, by which a tenant asks that the data is stored within a specific geographic region;
- Geographic exclusion, by which a tenant asks that the data is not stored within a geographic region;
- Security, such as encryption of data with a specific algorithm;

- Data dispersion, which defines the minimum geographic distance that data centers holding replicas of the data should have.

These terms are automatically transformed to system requirements. For example a durability of 99.999% can be translated to creating 3 replicas of a given container. The management directives are thereafter added as metadata to containers during their creation process and are used for optimizing their placement.

In order to efficiently manage the platform and make decisions, a clear and coherent global view of the system is needed. For this reason, VISION Cloud makes use of a monitoring mechanism that is able to collect, aggregate and distribute monitoring information (events) across the decision making components (that exploit a unified management model [48]) of the collaborating clusters and data centers of the Cloud. The metrics collected form the basis for QoS management, while the efficiency of the mechanism helps lower the total cost of ownership (TCO) of the system. The events are generated by different VISION Cloud components (sources) in response to – directly or indirectly – a user action (e.g, putting a new object), or periodic performance measurements (CPU load, network traffic, etc.). In order to efficiently use network resources (thus lowering TCO), and distribute as much as possible the needed computations, aggregation rules are applied on three levels of the platform, namely node, cluster and cloud level. Moreover, different fault-tolerance levels are used based on the importance of the generated events. Some events, needed for accounting and compliance reasons, are persisted and replicated while others are distributed on a best-effort basis.

The coupling of management metadata with an advanced monitoring framework also allows for proactive SLA violation detection schemes to be developed. Therefore, trends and patterns are continuously calculated on metrics defined in the SLAs. Building on the knowledge that is accumulated though the analysis of monitored parameters, the system is able to proactively detect conditions that could lead to degradation of the QoS delivered and take necessary management actions to avoid a possible SLA violation. For instance, the system can detect that a container is being increasingly accessed, leading the throughput offered to fall below the requested threshold, and triggering the replication of the container.

Moreover, in such a highly distributed environment multiple hardware faults are bound to happen concurrently. Therefore, VISION Cloud employs resiliency mechanisms to continuously check for faults and take recovery actions taking into consideration that faults may also affect the recovery process itself.

4.5.3 Summary

The amount of data being produced and stored in the Cloud is constantly increasing. VISION Cloud aims at providing an advanced storage cloud solution overcoming limitations of today's commercial offerings such as data lock-in, rudimentary SLAs, separation between compute and storage resources and security. Research is thus conducted, amongst others, in fields such as defining new data and compute models, SLA management and QoS provision, monitoring of large-scale infrastructures, security and compliance mechanisms. With respect to dependability, the aforementioned work on SLA management as well as resiliency mechanisms (e.g., through a component for fault-handling of containers and storage objects) will allow for the provision of an infrastructure and environment on specific levels of quality.

5. CONCLUSIONS

This paper presented some selected contributions of five ongoing EU-funded projects that tackle Cloud dependability issues. As described, each one of these projects has its own vision on how to extend and improve Cloud computing as we know it today. While there have been already a lot of exciting results there will be another wave of contributions during the next year which builds the final phase for most of these projects. All of them publish their results and typically provide source code for key components on their web pages [2, 4, 8, 10, 11]. The latter also offers a good stating point to get in touch and exchange ideas.

6. ACKNOWLEDGMENTS

We thank Christian Cachin and Tobias Distler for comments on a preliminary version of the paper and all projects partners that contributed to the works described in this paper. This work was partially supported by the European Commission through grants FP7-ICT-2009-5-257784 (Cloud-TM), FP7-ICT-2009-5-257438 (Contrail), FP7-ICT-2009-5-256910 (mOSAIC), FP7-ICT-2009-5-257243 (TClouds), FP7-2009-5-ICT-257019 (VISION Cloud).

7. REFERENCES

[1] BFT-SMaRt Web Site. `http://code.google.com/p/bft-smart/`.

[2] CloudTM Web Site. `http://www.cloudtm.eu/`.

[3] Contrail code repository. `http://contrail.projects.ow2.org/xwiki/bin/view/Main/`.

[4] Contrail web site. `http://contrail-project.eu/`.

[5] Cooperation themes in Horizon 2020. `http://ec.europa.eu/research/horizon2020/pdf/press/fact_sheet_fp7_themes_in_h2020.pdf`.

[6] Horizon 2020. `http://ec.europa.eu/research/horizon2020/index_en.cfm?pg=home&video=none`.

[7] mOSAIC code repository. `http://bitucket.org/mosaic`.

[8] mOSAIC Web Site. `http://www.mosaic-cloud.eu`.

[9] Open Cloud Computing Interface. `http://occi-wg.org`.

[10] TClouds Web Site. `http://www.tclouds-project.eu/`.

[11] VISION Cloud Web Site. `http://www.visioncloud.eu/`.

[12] Open Virtualization Format Specification, Jan 2010. DMTF Standard. `http://dmtf.org/standards/ovf`.

[13] M. K. Aguilera, A. Merchant, M. Shah, A. Veitch, and C. Karamanolis. Sinfonia: A new paradigm for building scalable distributed systems. *ACM Transactions on Computer Systems*, 27(3), 2009.

[14] K. R. Balraj. An Approach to Achieve Delegation of Sensitive RESTful Resources on Storage Cloud. In *2nd*

Workshop on Software Services: Cloud Computing and Applications based on Software Services, 2011.

[15] H. Berenson, P. Bernstein, J. Gray, J. Melton, E. O'Neil, and P. O'Neil. A critique of ANSI SQL isolation levels. In *Proceedings of the International Conference on Management of Data (SIGMOD '95)*, pages 1–10, 1995.

[16] A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa. DepSky: Dependable and Secure Storage in a Cloud-of-Clouds. In *Proceedings of the 6th EuroSys Conference (EuroSys '11)*, pages 31–46, 2011.

[17] S. Bleikertz, T. Groß, M. Schunter, and K. Eriksson. Automated Information Flow Analysis of Virtualized Infrastructures. In *Proceedings of the 16th European Symposium on Research in Computer Security (ESORICS '11)*, pages 392–415, 2011.

[18] S. Bugiel, T. Pöppelmann, S. Nürnberger, A.-R. Sadeghi, and T. Schneider. AmazonIA: When Elasticity Snaps Back. In *Proceedings of the 18th Conference on Computer and Communications Security (CCS '11)*, pages 389–400, 2011.

[19] S. Cabuk, C. I. Dalton, K. Eriksson, D. Kuhlmann, H. V. Ramasamy, G. Ramunno, A.-R. Sadeghi, M. Schunter, and C. Stüble. Towards automated security policy enforcement in multi-tenant virtual data centers. *Journal of Computer Security*, 18(1):89–121, 2010.

[20] Contrail Consortium. Design of the virtual infrastructure network, 2011. Contrail Deliverable - D4.1.

[21] M. Coppola, P. Dazzi, A. Lazouski, F. Martinelli, P. Mori, J. Jensen, I. Johnson, and P. Kershaw. The Contrail approach to cloud federations. In *Proceedings of the International Symposium on Grids and Clouds (ISGC '12)*, 2012.

[22] P. Costa, M. Pasin, A. Bessani, and M. Correia. Byzantine fault-tolerant MapReduce: Faults are not just crashes. In *Proc. of the 3rd Int. Conference on Cloud Computing Technology and Science (CloudCom'11)*, 2011.

[23] M. Couceiro, P. Romano, and L. Rodrigues. PolyCert: Polymorphic Self-optimizing Replication for In-Memory Transactional Grids. In *Proceedings of the 12th Conference on Middleware (Middleware '11)*, pages 309–328, 2011.

[24] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: Amazon's highly available key-value store. In *Proceedings of the 21st Symposium on Operating Systems Principles (SOSP '07)*, pages 205–220, 2007.

[25] B. Di Martino, D. Petcu, R. Cossu, P. Goncalves, T. Mahr, and M. Loichate. Building a Mosaic of Clouds. In *Proceedings of the Euro-Par 2010 Parallel Processing Workshops (Euro-Par '10 Workshops)*, pages 571–578, 2011.

[26] D. Didona, P. Romano, S. Peluso, and F. Quaglia. Transactional Auto Scaler: Elastic scaling of NoSQL transactional data grids. In *Proceedings of the 9th International Conference on Autonomic Computing (ICAC '12)*, 2012.

[27] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg. Proofs of ownership in remote storage systems. In *Proceedings of the 18th ACM conference on Computer and communications security (CCS'11)*, pages 491–500, 2011.

[28] P. Harsh, Y. Jegou, R. G. Cascella, and C. Morin. Contrail virtual execution platform: Challenges in being part of a cloud federation. In *Proceedings of the 4th European Conference Towards a Service-based Internet (ServiceWave '11)*, pages 50–61, 2011.

[29] P. Harsh, Y. Jegou, R. G. Cascella, and C. Morin. Open computing infrastructures for elastic services. In *European Research Activities in Cloud Computing*. Cambridge Scholars Publishing, 2012.

[30] F. Hupfeld, T. Cortes, B. Kolbeck, J. Stender, E. Focht, M. Hess, J. Malo, J. Marti, and E. Cesario. The XtreemFS architecture – a case for object-based file systems in grids. *Concurrency and Computation: Practice & Experience*, 20(17):2049–2060, 2008.

[31] R. Kapitza, J. Behl, C. Cachin, T. Distler, S. Kuhnle, S. V. Mohammadi, W. Schröder-Preikschat, and K. Stengel. CheapBFT: Resource-efficient Byzantine Fault Tolerance. In *Proceedings of the 7th EuroSys Conference (EuroSys '12)*, pages 295–308, 2012.

[32] E. K. Kolodner, S. Tal, D. Kyriazis, D. Naor, M. Allalouf, L. Bonelli, P. Brand, A. Eckert, E. Elmroth, S. V. Gogouvitis, D. Harnik, F. Hernandez, M. C. Jaeger, E. B. Lakew, J. M. Lopez, M. Lorenz, A. Messina, A. Shulman-Peleg, R. Talyansky, A. Voulodimos, and Y. Wolfsthal. A Cloud Environment for Data-intensive Storage Services. In *Proceedings of the 3rd International Conference on Cloud Computing Technology and Science (CloudCom'11)*, pages 357–366, 2011.

[33] A. Lakshman and P. Malik. Cassandra: A decentralized structured storage system. *SIGOPS Operating Systems Review*, 44:35–40, 2010.

[34] F. Marchioni and M. Surtani. *Infinispan Data Grid Platform*. PACKT Publishing, 2012.

[35] N. Mavrogeorgi, S. V. Gogouvitis, A. Voulodimos, G. Katsaros, S. Koutsoutos, D. Kyriazis, T. Varvarigou, and E. Salant. Content Based SLAs in Cloud Computing Environments. In *5th International Conference on Cloud Computing*, 2012.

[36] S. Peluso, P. Ruivo, P. Romano, F. Quaglia, and L. Rodrigues. When Scalability Meets Consistency: Genuine Multiversion Update-Serializable Partial Data Replication. In *Proceedings of the 32nd International Conference on Distributed Computing Systems (ICDCS '12)*, 2012.

[37] D. Petcu. Invitation to a Journey in the ERA of Cloud Computing. In *European Research Activities in Cloud Computing*, pages 1–18. Cambridge Scholars Publishing, 2012.

[38] D. Petcu, M. Frincu, C. Craciun, S. Panica, M. Neagul, and G. Macariu. Towards Open-Source Cloudware. In *Proceedings of the 4th International Conference on Utility and Cloud Computing (UCC '11)*, pages 330 –331, 2011.

[39] D. Petcu, G. Macariu, S. Panica, and C. Crăciun. Portable Cloud applications – From theory to practice. *Future Generation Computer Systems*, 2012. doi:10.1016/j.future.2012.01.009.

[40] D. Petcu and J. Vázquez-Poletti (Editors). *European Research Activities in Cloud Computing*. CSP, 2012.

[41] G. Pierre, I. el Helw, C. Stratan, A. Oprescu, T. Kielmann, T. Schuett, J. Stender, M. Artac, and A. Cernivec. ConPaaS: An integrated runtime environment for elastic cloud applications. In *Proceedings of the 12th Conference on Middleware – Posters and Demos Track (Middleware '11)*, pages 5:1–5:2, 2011.

[42] P. Romano, L. Rodrigues, N. Carvalho, and J. Cachopo. Cloud-TM: Harnessing the cloud with distributed transactional memories. *SIGOPS Operating Systems Review*, 44(2):1–6, 2010.

[43] L. Schubert and K. Jeffery. Advances in Clouds. Research in Future Cloud Computing. Expert Group Report, 2012. `http://cordis.europa.eu/fp7/ict/ssai/docs/cloud-report-final.pdf`.

[44] N. Shavit and D. Touitou. Software transactional memory. In *Proceedings of the 14th Symposium on Principles of Distributed Computing (PODC '95)*, pages 204–213, 1995.

[45] S. Venticinque, R. Aversa, B. Di Martino, M. Rak, and D. Petcu. A Cloud Agency for SLA Negotiation and Management. In *Proceedings of the Euro-Par 2010 Parallel Processing Workshops (Euro-Par '10 Workshops)*, pages 587–594, 2011.

[46] P. Verissimo, A. Bessani, and M. Pasin. The TClouds Architecture: Open and Resilient Cloud-of-clouds Computing. In *Proceedings of the 2nd Workshop on Dependability of Clouds, Data Centers and Virtual Machine Technology (DCDV '12)*, 2012.

[47] G. Veronese, M. Correia, A. Bessani, and L. Lung. Efficient Byzantine Fault Tolerance. *IEEE Transactions on Computers*, 2012.

[48] A. Voulodimos, S. V. Gogouvitis, N. Mavrogeorgi, R. Talyansky, D. Kyriazis, S. Koutsoutos, V. Alexandrou, E. Kolodner, P. Brand, and T. Varvarigou. A Unified Management Model for Data Intensive Storage Clouds. *Proceedings of the Symposium on Network Cloud Computing and Applications (NCCA'11)*, pages 69–72, 2011.