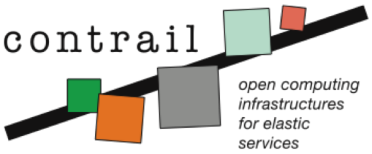
# Contrail final publishable summary report

Christine Morin, Roberto Cascella





Project no. 257438

# CONTRAIL

Integrated Project
OPEN COMPUTING INFRASTRUCTURES FOR ELASTIC SERVICES

## Final publishable summary report

Version 1

Contrail consortium

April $24^{th}$, 2014

| **Project co-funded by the European Commission within the Seventh Framework Programme** | | |
|---|---|---|
| **Dissemination Level** | | |
| **PU** | Public | √ |
| **PP** | Restricted to other programme participants (including the Commission Services) | |
| **RE** | Restricted to a group specified by the consortium (including the Commission Services) | |
| **CO** | Confidential, only for members of the consortium (including the Commission Services) | |

## Executive summary

This document comprises the final report on the Integrated Project Contrail - “Open Computing Infrastructures for Elastic Services". The project started in October 2010 and ended in January 2014. The Contrail software stack provides for cloud administrators a platform to manage federated heterogeneous resources offering customers advanced cloud services by selecting the most suitable cloud provider matching the application requirements with performance and security guarantees. Contrail gives the illusion of using a single cloud provider which can dynamically adapt to the customers’ needs. Cloud consumers can now use a trustworthy platform to run their applications.

We have developed a comprehensive set of cloud services that can be exploited by means of the Contrail software stack or as independent components. Contrail software components range at different levels from PaaS to federation of IaaS clouds, and IaaS resources. The main Contrail services are: single sign on and federated identity, automatic SLA negotiation and provider selection, dynamic authorisation and access control, distributed application deployment under SLA constraints, reliable distributed storage, virtual networks, monitoring and accounting, and PaaS runtime services.

The Contrail software has been experimented and validated with a wide range of applications, both scientific and user oriented. One demonstrator was implemented, shown at different events and available on the web. The project results are available as open source software. Some of the results are already exploited by other projects and the consortium member organizations plan to exploit some of the results in follow-up research projects and in future products.

# Contents

# 1 The challenge

After decades in which companies used to host their entire IT infrastructures in-house, a major shift is occurring where these infrastructures are outsourced to external operators such as data centers and computing clouds. The growth of interest toward computing clouds is facilitated by virtualization technologies which offer several advantages over traditional data center approach to computing. On the one hand, companies can move their applications to the cloud freeing themselves from the control and management of the infrastructure so that they can focus on the deployed services. On the other hand, companies can rent resources of cloud providers only when needed according to a pay-as-you-go pricing model reducing the cost of the infrastructure. In a nutshell, this paradigm represents a new opportunity for companies and organisations to rely on highly dynamic distributed infrastructures to run applications and services. The advantages of cloud computing are well known: on-demand self-service; broad network access; resource pooling to optimize usage of provider resources; rapid elasticity to dynamically adapt the allocated resources to the customer needs; and measured service to provide performance guarantees and support the pay-per-use model [22].

The cloud computing market is in rapid expansion and many cloud providers are flourishing in Europe to offer Infrastructure as a Service (IaaS) services to contrast big players, like Amazon, that have so far dominated. The market opportunities are quite challenging for new IaaS cloud providers, which might have limited resources to offer. They play in a competitive service market where the organisations and companies they want to attract are looking for large pool of resources, as well as for guarantees in terms of the reliability and availability for their services.

However the growth of cloud computing may soon be hindered by other factors such as the customers' concerns to be locked-in within a single commercial offer (which reduces the necessary competition between many infrastructure providers) and the lack of performance and cost predictability of current clouds. Moreover, the fact that many cloud commercial offers exists makes very difficult to select the right cloud service provider because of the increasing variety of services and their different pricing schemes. Users are not familiar with paying per application execution and costs are difficult to estimate because of the lack of clear cost models that account for processing power, storage capacity, data transfers from the cloud to the outside. As a result, running an application in the cloud may be expensive if the number and size of VMs are not properly selected according to the real fluctuating application needs. This adds to the complexity of switching from one provider to another and managing multiple accounts for different type of services.

Other major issues are legal requirements for data: they cannot be stored anywhere for legal jurisdiction implication or need to have specific privacy requirements to stick with company or country legislation. Security of the infrastructure is another major challenge as well as ownership and privacy issues of the data stored in the cloud. In fact, potential customers will adopt cloud computing only if they feel confident in the security of the computing environment. Security support is a key factor because business users' data could have a high economic value and data theft or damage could result in considerable financial and repetitional loss.

These concerns can be summarised in the lack of trust on the clouds with customers being sceptical in the cloud model and in services offered by a cloud provider if no guarantees exist. The Contrail project addresses all these challenges developing dependable and interoperable cloud services that customers can trust and easily use in the open cloud market. Contrail provides services to federate IaaS clouds, SLA management in federated clouds, and an extensible PaaS for easy deployment of a wide range of elastic applications. All services and infrastructure are secured and monitored to guarantee dependability.

# 2 Addressing the challenge

Bringing trust to the Cloud is the key for a mass adoption of the cloud computing paradigm. The vision of the Contrail project [1, 4] is that a consumer should be able to trust the Cloud with the data and applications while at the same time being relieved from the complexity to deploy distributed applications on the Cloud. This is the first step to allow open access to shared computing resources, where the resources that belong to different operators will be integrated into a single homogeneous federated Cloud that users can access seamlessly.

## 2.1 Fundamental concepts

The goal of the Contrail project is to design, implement, evaluate and promote an open source system for Cloud federations. The design of Contrail has been guided by two fundamental concepts: modularity and transparency. Modularity is achieved via clear specification of internal APIs so that existing software could be reused and most important Contrail components could leave independently avoiding that monolithic approach could mine potential opportunities. Transparency and ease of usage for external users are achieved via the federation services that hide the complexity of dealing with heterogeneous resources.

These concepts imply three essential properties that are the main challenges identified in previous section and that will be addressed in the remaining of this section:

**Interoperability** federating heterogeneous cloud providers and complying with all major standards;
**Dependability** providing reliability and high availability through SLAs, and ensuring federated identity management and trust in the cloud platforms;
**Transparency** reducing the complexity of deploying applications over heterogeneous Cloud providers.

The key challenges Contrail addresses in existing commercial and private/community Clouds are: the lack of standardised rich and stable interfaces; limited trust from customers; and relatively poor Quality of Service (QoS) guarantees regarding the performance and availability of Cloud resources. Addressing these important issues is fundamental to support large user communities formed of individual citizens and/or organisations relying on Cloud resources for their mission-critical applications.

## 2.2 Contrail properties

Contrail [1] is a software stack that enables a federation of heterogenous clouds to deploy distributed applications under QoS and QoP constraints. Contrail is an open source integrated approach to virtualization, which aims at offering Infrastructure as a Service services (IaaS), services for federating IaaS clouds, and Contrail Platform as a Service services (ConPaaS) on top of federated clouds.

**Vendor lock-in.** A first step to address the users' concerns is to avoid vendor lock-in giving the opportunity to select the most convenient cloud provider based on the application requirements or price of the offer. Cloud lock-in makes the deployment of applications on different providers difficult for users. This complexity comes from the fact that users must customize the application and its Virtual Machine Images (VMI) to a specific Infrastructure-as-a-Service (IaaS) type used by cloud providers. When the user wants to switch from a cloud provider to another, she has to repeat this customization, making difficult to escape vendor lock-in and limiting the *portability* of the application. However, IaaS cloud interfaces and management software are evolving, making the repetition of the customization step and the deployment of portable application hard to achieve.

**Interoperability.** Interoperability among cloud providers is the only way to challenge vendor lock-in and open the way toward the defragmentation of the market and more competitive offers. Moreover, interoperability is a need for small players to enter a market dominated by big cloud providers which can count on a huge number of resources. As such, interoperability becomes even more handy and needed in a multi provider scenario, where customers can protect their investment by counting on a wider number of options to offer their services on top of cloud systems. At the same time, they take full advantage of the elasticity and pay-as-you-go concepts. One way to achieve interoperability is via the adoption of **cloud standards**. Many standards are now flourishing to facilitate interoperability across different cloud providers and improve the portability of applications. The European Commission has seen the need to coordinate the jungle of standards to avoid the fragmentation of the digital market with the creation of a new Cloud standards co-ordination Task Group within ETSI [2]. Another approach is to deploy a middleware service that adapts the application to a specific cloud provider. A more comprehensive way to address interoperability is the cloud federation: it can help in hiding the complexity of managing heterogeneous resources and using a multitude of cloud providers at the same time.

The Contrail project follows this approach and fusions them under the umbrella of a cloud federation which supports heterogeneous cloud providers via the middleware services of the Virtual Execution Platform (VEP) component. The Contrail Federation is the only entry point of the customers who negotiate their application directly with the Federation which presents the offer of several Cloud providers. Standards are in use in Contrail to provide a unique framework to describe the applications via the Open Virtualization Format (OVF by DMTF [11]) standard.

**Dependability.** However, on top of these federated cloud providers it is of utmost importance to ensure the availability of the computing resources and to provide strict guarantees in terms of quality of service (QoS) and quality of protection (QoP). Offering dependable services on top of an unreliable physical infrastructure is both a challenging task and an important feature to make cloud computing attractive for users and organisations. Hence, the cloud provider is not only charged with the complexity of managing a potentially distributed infrastructure and dispatching the virtual machines (VMs) of the applications to the physical resources, but with the need to offer guarantees to customers transparently. Users and organizations should have the opportunity to specify these features in a negotiated Service Level Agreement (SLA) and to monitor them at runtime. Deploying applications and services under a SLA will make cloud computing a valid alternative to private data centers responding to the users requirements in terms of availability, reliable application execution, and security.

**Transparency.** In Contrail, the user is relieved from the complexity of managing the access to individual cloud providers and can focus on specifying the service or application to be automatically deployed over a multitude of heterogeneous providers. The providers can rely on different cloud technologies, exploit different hardware, or offer different types of guarantees. In a nutshell, Contrail simplifies the whole process from cloud service provider selection to application deployment and monitoring.

## 2.3 Contrail services

Contrail offers performance (QoS) and security (QoP) guarantees via SLA enforcement by monitoring the execution of the application, and a scalable management of the computing resources via an interoperable federation. The federation service is the interface with the user, who needs to submit the description of the distributed application to be deployed in the cloud, along with its runtime configuration, and specify

the requirements in terms of OVF (Open Virtualization Format specification) [11] and SLA documents respectively. Then, the federation ensures that the providers' resources are utilized as needed for offering a dependable and trustworthy cloud service to customers.

The application is deployed via the Virtual Execution Platform (VEP), a provider-level service supporting federation-level interoperability. The use of VEP allows to deploy a distributed application under the terms of a SLA over the resources of any of the supported IaaS providers, regardless e.g. of the underlying cloud management system. Elasticity of the application is also ensured by monitoring the usage of the resources stated in a negotiated SLA, both within the cloud provider infrastructure and at the federation level.

Contrail implements a number of other services to address dependability issues. Contrail provides a reliable and highly available storage service, named Global Autonomous File System (GAFS), based on XtreemFS. Contrail uses GAFS to store VM images and system logs, but GAFS also serves to provide scalable Storage as-a-Service to cloud users and applications. Most important is the possibility to specify the level of protection of the stored data and the location of the storage due to specific legal requirements. These requirements are critical to make the provider resources trustworthy, making cloud computing suitable to run the users' businesses safely.

Contrail technology also deploys ConPaaS services [26], which are self-managed, elastic, and scalable. A ConPaaS service can deploy itself on the cloud, monitor its own performance, and increase or decrease its processing capacity by dynamically (de-)provisioning instances of itself in the cloud. The tight integration with the Federation ensures that a ConPaaS service can be deployed over different cloud providers to guarantee elasticity within the constraint of the negotiated SLA, which specifies the QoS terms. Hence, ConPaaS services integrates security and availability guarantees for a reliable execution.

Dependability in Contrail is completed with the ConSec (Contrail Security) services. The security of the federated cloud infrastructure is important as it increases the range of work that can be entrusted to the Cloud without being locked in to a single provider. Arguably, without cloud security, there can be no federated access to Clouds. All services in Contrail are adequately protected against unauthorized access or modification. Authentication and authorization are two essential characteristics to identify users and protect their applications and data from others or even the cloud provider itself.

ConSec provides solutions to manage federated identities and integrates single-sign-on and identity providers (Shibboleth and OpenId) to simplify the authentication process. Delegation is also important to enable services to act on behalf of users temporarily. Contrail has chosen to use X.509 certificates for user credentials and OAuth2 to delegate these credentials, and adopts mechanisms to manage the trust in Cloud Providers for delegations. This brings us a system which enables to delegate certificates for any purpose. Finally, the authorisation system integrates mechanisms for dynamic usage control with online verification of attributes during the application runtime.

# 3 Contrail achievements toward a trustworthy cloud

Figure 1 depicts the Contrail architecture. The federation layer is the entry-point for users, who register and authenticate to use the Contrail services. The way the Contrail federation is conceived enables seamless access to the resources of multiple cloud providers, avoiding potential vendor lock-in for the end users. It reaches a high degree of interoperability by managing private or public cloud providers' resources regardless of the technology implemented or underlying hardware. Contrail provides an infrastructure for federated identity management which makes cloud federation possible by allowing users to securely authenticate to the Contrail Federation service either by using Contrail accounts or external identity providers (IdPs), using

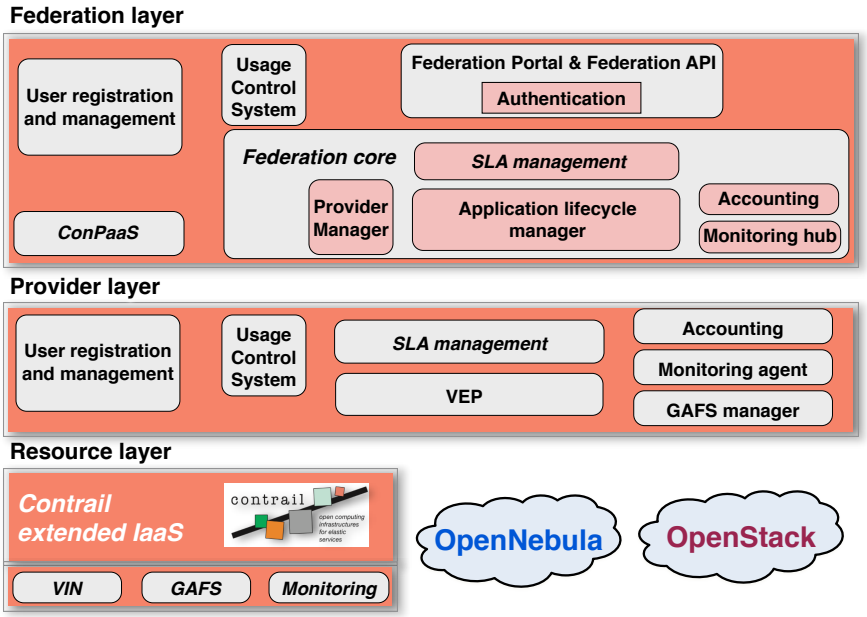



Figure 1: Simplified vision of the Contrail architecture

either OpenId or Shibboleth.

The Contrail federation enables users to deploy distributed applications on demand on different cloud providers by only interacting with this single component. The federation incorporates in the *Federation core* the necessary functionalities to negotiate SLAs and monitor their enforcement while the application is running. The Federation core, together with its interfaces, is deployed at each *Federation Access Point* (FAP). A Contrail federation is thus made up of distributed, interconnected instances of FAPs and providers. The user submits the description of the application based on the OVF [11] standard format and negotiates the SLAs terms, then the federation selects the most suitable cloud providers based on the resources available, the expressed SLA terms for QoS and QoP, and the reputation of the providers, i.e., matching the level of performance and trustworthiness required by the application. Hence, the federation proceeds to negotiate proper SLA terms with each provider in a transparent way for the users.

Contrail technology is able to satisfy the user needs for the deployment of elastic and scalable applications by adding or removing resources in order to satisfy the SLA terms without the need of a new SLA negotiation. Monitoring and auditing are performed during application execution, to detect any violation of the SLA.

Proper authorization and security mechanisms are enforced primarily at the federation layer and then at the other layers to guarantee quality of protection (QoP).

The provider layer implements the business part of a cloud provider: (i) negotiation with the federation and enforcement of the SLA; (ii) resource reservation and application management; (iii) monitoring and accounting. The resource layer is in charge of managing the physical resources of a cloud provider. Contrail does not implement a new IaaS, but leverages the existing ones[1] by adding those functionalities required to provide performance and security guarantees for an application.

In Contrail, each cloud provider runs a copy of the Virtual Execution Platform (VEP) software which in turn seamlessly integrates the provider resources within the Contrail federation. VEP is an open source technology implementing standards that exploits resource virtualization to provide virtualized distributed

[1]Contrail extends and supports OpenNebula and OpenStack. The support of non *Contrail extended IaaS* limits the level of control of the resources, thus the type of guarantees that could be offered to a customer.

infrastructures for the deployment of end-user applications independently from the underlying platform: Contrail extended IaaS, OpenNebula or OpenStack in the current implementation. It offers an application deployment platform, which is resilient to operational failures and which ensures that an application is deployed respecting QoS requirements. The degree of interoperability and features that the federation can exploit on each single cloud provider depend on the specific functionalities implemented at the cloud provider level. Interoperability is achieved through the VEP component.

The Contrail resource layer integrates the services of the Virtual Infrastructure Network (VIN) component which assures the internetworking between Virtual Machines (VMs) of an application and with the public Internet, providing bandwidth reservation capabilities within a data center and isolated environments for an application. VIN implements Virtual Private Networks (VPNs) and provides network tunnels that are either encrypted using IPSec, or unencrypted using GRE. VIN provides connectivity for federated cloud resources, supporting different cloud infrastructure stacks, in combination with external GAFS file servers and client machines.

The Global Autonomous File System (GAFS) provides the storage resources for most kinds of data in the system. On the one hand, it is used to store infrastructure-related data, such as virtual machine images and system logs. On the other hand, it provides scalable Storage-as-a-Service to Cloud users and applications, and offer users the possibility to host their own storage services.

On top of the federation, a Contrail customer can use the Contrail PaaS (ConPaaS) services. Services are designed to be composable and each service is self-managed and elastic. ConPaaS services can be used independently of Contrail and can run on multiple IaaS infrastructures. ConPaaS currently contains nine services:

- Two Web hosting services respectively specialized for hosting PHP and JSP applications;
- MySQL database service;
- Scalarix service offering a scalable in-memory key-value store;
- MapReduce service providing the well-known high-performance computation framework;
- TaskFarming service for high-performance batch processing;
- Selenium service for functional testing of web applications;
- XtreemFS service offering a distributed and replicated file system;
- HTC service (for high-throughput computing) providing a throughput-oriented scheduler for bags of tasks submitted on demand.

The following sections discuss all Contrail services in details. We start with the Contrail components that enable the deployment of distributed and dependable applications under the terms of a SLA over a federation of heterogeneous cloud providers. These are (i) the Contrail Federation integrating under a common umbrella the resources of different cloud providers relieving the user from the negotiation of the application with each provider (see Section 3.1; (ii) the SLA component running within the federation core and at the provider layer offering SLA negotiation, management, and enforcement services thanks to the monitoring services (see Section 3.2); (iii) the security solutions integrated at all levels of the Contrail software stack (see Section 3.3); and (iv) the services to manage the virtual resources of a cloud provider and offering the deployment of elastic and isolated application with reliable storage support within the constraints expressed in a SLA (see Section 3.4). Then, we present the Contrail PaaS services in Section 3.5 and in Section 3.6 the use cases that have been chosen to represent a diverse set of applications, end-user communities, architectural scenarios and infrastructure requirements. All of the use cases contain real-world applications with user bases able to evaluate the end user experience delivered by Contrail. Section 3.7 concludes this document drawing some future perspectives.

### 3.1 Contrail Federation

A cloud federation is in its essence a platform where multiple cloud providers interoperate with each other, creating a service marketplace for users to dynamically match their needs with the offers from a subset of providers.

The Contrail federation goes beyond this basic goal, in pursuing a *trusted* platform, where trust comes from a pervasive support for security, availability, reliability, and accountability.

Such being the goal, a key element of Contrail federation is to provide state-of-the-art SLA management of QoS and QoP. Contrail also removes most basic user lock-in constraints offering a uniform approach to actor identification, management policies, costs and application description.

In the last few years, the cloud market has grown in terms of its IT market penetration, of the number of players in cloud service provisioning and in terms of differentiation of services between the providers. This variegated cloud offer is an opportunity for companies that want to use public clouds for their applications, but also forces to deal with a non trivial comparison and selection problem.

As the cloud market was still taking shape, different proprietary protocols to describe and rent services did imply a certain degree of user lock-in. While on the one hand we now have much more options to choose from (at the IaaS as well as the PaaS and SaaS levels), on the other hand, and despite strong efforts toward standardization and interoperation [5, 11, 12, 24], it has become increasingly complex to choose between semantically equivalent services with different metrics of cost, performance, reliability, security and elasticity. Complexity arises (i) from the need to match the services, and the service quality level descriptions in different languages, with different protocols used to set up, monitor and steer them, as well as (ii) from the need to plan service utilization in order to optimize a user-specific trade-off of the aforementioned metrics, gathering and exploiting information about a multitude of service providers.

Cloud brokering is nowadays the rising approach to address those issues [31], with both open source (DeltaCloud [7, 10]) and commercial solutions (CloudSwitch, Enstratus, Rightscale, Kaavo) already being available. Cloud brokers target interoperation between one user application and one provider, only considering multiple cloud interconnection for the restricted case of cloudbursting from the user's private cloud to a public one. Besides, removing API-related lock-in barriers with respect to providers can result in tying up the user to the broker management interface.

The Contrail approach to cloud federations mainly focuses on provider-level integration of infrastructural services (IaaS) including a heterogeneous population of providers of computation, network, and storage services. As opposed to a broker-based approach, the end-user can exploit advanced inter-provider deployment and coordination, even for a single application, and benefit from state-of-the art federated security as well as federation-ubiquitous SLA mechanisms. These features constitute the basis for higher-level services (PaaS, SaaS) and allow to provide standardized guarantees to the platform user. While cloud brokering relies on and fosters a more competitive cloud resource market, Contrail federations also promote *cooperation* among several providers as a way to open new market opportunities, and fully addressing issues (i) and (ii) outlined before.

To this end, the Contrail federation [8] meets several design constraints. *Security* and *decentralization* are key ones: the many Federation Access Points (FAPs, Fig. 2) implementing the top layer of the Contrail architecture are available to any federation user and can interact with any provider within the federation. Beside leveraging state-of-the-art federated authentication solutions [16] and authorization mechanisms [20], the *Data Store* module of the federation has built-in mechanisms for synchronizing critical data among FAPs. The Data Store provides permanent memory of many information flows (user accounting, reputation monitoring, violation monitoring) each one with its own synchronization policy.

The FAPs constitute a network of broker-like access points which are peers on a common network,

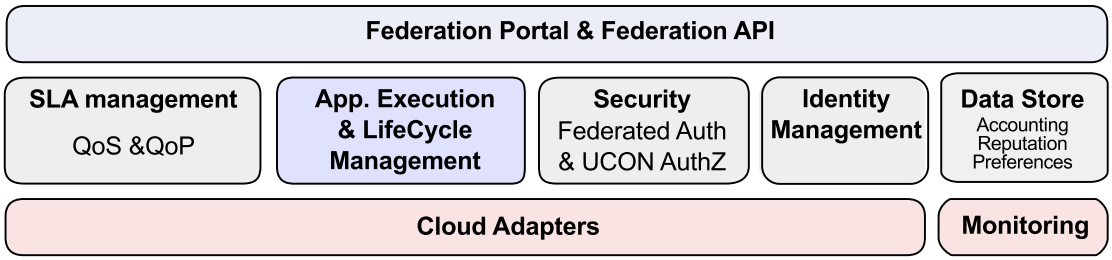



Figure 2: A Federation Access Point. Each FAP implements the topmost layer of the Contrail abstract architecture, including the Federation Core, its GUI and API.

whose underlying software infrastructure includes Contrail's security support.

Each FAP can then be co-located with a cloud provider, e.g. making the access to local resources faster or cheaper, but the federation (globally seen) and each one of its FAPs are free to choose resources for each application deployment request from any provider in the federation, according to both the user desired SLA and to the dynamically evolving conditions of all the federated service providers. The approach can easily work-around transient issues of network overload and service requests exceeding the availability at a specific provider. Besides, the approach also allows a complete separation of the interests of the cloud provider hosting a FAP from those of the federation and its users.

A key aim in designing the Contrail federation was to provide support for *efficient mapping* of applications under *complex SLA constraints*. The actual mapping of the application over one or more providers is done in the *Application Execution and LifeCycle Management* module (which includes the services of the *Application Lifecycle Manager* shown in Fig. 1). The mapping of applications, or part of them, on the available service providers is done on the ground of several desiderata:

- the user needs, expressed by the application description and its SLA proposal;
- information gathered by the federation about provider reputation (e.g. estimated reliability) and available resources;
- the negotiation carried on by the SLA management service, directly (via the GUI) or indirectly (via preferences) driven by the user.

The *SLA management* module inside the FAP architecture is in charge of carrying the topmost level of the hierarchical SLA negotiation, as described in Section 3.2, trying to achieve SLA contracts with one or more providers that can overall satisfy the user-proposed SLA. While the main process of SLA negotiation as implemented by the SLA management modules is more thoroughly described in the next section, the federation support in each FAP is in charge of a few important activities that ensure the scalability and reliability of the overall approach.

A first key point is that the Contrail monitoring subsystem provides continuous information about each provider behaviour to the FAPs currently using it. Feedback about SLA violations and related events, service failures or the amount of applications active at a given provider, allows a FAP to estimate the current reliability of that provider. Different FAPs may also exchange their esteems with one other to gather more information.

While in principle any provider within a Contrail federation can provide resources to any given application, holding a hierarchical SLA negotiation reaching all of them is obviously unpractical. Contrail federation chooses instead a fully scalable approach, narrowing down each negotiation to a subset of providers which are known to potentially match static constraints in the SLA. The federation further instructs the SLA

management module to contact those providers which rank the best with respect to the user preferences and to the reliability estimates of the current FAP.

Application mapping exploits a set of heuristics in order to optimize the user-defined trade-off among application metrics, e.g. balancing economic cost and performance levels. To simplify this task, we employ a software toolkit which translates different parts of the application description (namely OVF files, SLA, deployment information) to and from several standard formats and into an in-memory graph representation. This approach is essential in order to allow the Contrail federation to support applications that exploit more than one provider at the same time. Graph-based optimization algorithms can be applied to achieve this aim, and the whole application structure can be modified, decomposed and translated in many ways to allow for composite deployment over possibly different cloud providers and SLA contracts.

Contrail directly supports the deployment of a single application on top of distinct providers for computation, storage and network services. The integrated management of all deployment constraints in Contrail federations allows for this initial case of application decomposition. Different and more general cases of application distribution have been designed, and are possible via dedicated heuristics which tackle specific application structures and decompose them along with their associated SLA.

The system is designed to also monitor and control the application execution, to possibly perform resource migration or elasticity management.

Finally, achieving strong *Interoperability* and code flexibility was obviously a main issue in the federation architecture design. Toward the user, a classical REST approach has been followed, with tools that allow accessing the federation services via browser and command line. A great deal of features are made easy since the VEP, presented in Section 3.4.1, shields the Contrail federation from the specificities of providers about local deployment, monitoring, and SLA management. The FAP, implementing the operational functionalities of the Federation core (shown in Fig. 1), thus has the task to coordinate (via an extendable set of *cloud adapters*) different federation entities (one or more VEP instances, VIN and GAFS resources) for the sake of a specific application.

## 3.2 Contrail SLA management

A Service Level Agreement (SLA) is the part of a contract between two parties (the provider and the consumer) about the usage of some service. The service itself is defined in the SLA along with the guarantees offered by the provider. SLAs can be a differentiator for cloud providers in the eyes of customers looking for services characterized not only by their low price but also by their quality.

The scope of work performed in Contrail covered in particular Infrastructure as a Service Clouds and ranges from supporting autonomic SLA Management for different types of single Cloud providers to implementing the core functionalities for SLA Management in Cloud federations.

Contrail SLA Management has the objective to build the technology for managing SLAs in Clouds. This objective could be formulated according to many aspects, from specification and modeling to life-cycle definition, from negotiation to resource planning, from monitoring Quality of Services (QoS) to enforcing it. SLAs in Contrail have also been used to express aspects of the Quality of Protection (QoP) offered by Cloud providers. To cope with so many aspects, Contrail builds on existing research and technology and reuses as much as possible from previous research projects. The major source of reuse for SLAs is the FP7 research project SLA@SOI, from which Contrail inherited the base framework and the SLA syntax model, customizing and extending them to cope with the identified use case requirements and research objectives.

Contrail SLA Management provides components for handling SLAs and SLA Templates, for negotiating SLAs and ensuring SLA compliance of provisioning requests, for monitoring and enforcing terms and for resource usage accounting.

The implementation of SLAs for Cloud Computing introduces some challenges, which can be summarized as follows:

- identification of SLA terms;
- connection of SLA terms with the provided services;
- automatic/autonomic support for the full life cycle of SLAs;
- extension of SLAs to Quality of Protection (QoP);
- SLA support for elasticity;
- SLAs for Cloud federations.

In the following we discuss the challenges and the solutions defined in Contrail. A first challenge is related to the identification and classification of SLA terms. SLA terms should be monitorable and enforceable, should be supported by the underlying Cloud implementation and, above all, should be meaningful to users. The major SLA terms supported by Contrail are represented in Table 1.

QoS terms for IaaS providers can either express guarantees on the whole infrastructure, such as *availability*, or guarantees on a single resource, such as the *cpu_speed* of a specific VM.

Some innovative SLA terms that are supported in the last release of Contrail are: *co_locate_rack*, *not_co_locate_host* and *minimum_LoA*. *Co_locate_rack* applied to a single VirtualSystem requests that all the VMs instantiated from it are scheduled on hosts that are part of the same rack, thus likely improving the speed of the network among them. *Not_co_locate_host*, also applied to a single VirtualSystem, specifies that all the VMs instantiated from that VirtualSystem must be scheduled on different hosts, thus paving the way for high availability configurations. The new *minimum_LoA* is a QoP term that specifies the minimum Level of Assurance (in authentication) that the resource requires, or the user defines for others to access their resource. Sensitive data will require a higher LoA: the federation authentication code sets the user's LoA when they authenticate; the required minimum is set at negotiation time and is also transmitted as an attribute when the authorization checks run. Access to resources is blocked if $LoA < minimum\_LoA$.

The support of security related SLA terms (defined in Table 1) requires the extension of SLAs to Quality of Protection. This involves, as for QoS, identifying the right and meaningful SLA terms, but also making sure that their implementation is feasible and identifying the right architecture and design for making them work. Geographic resource location and support for Level of Assurance (LoA) have been the most important QoP results obtained by Contrail.

| **SLA term** | **Description** |
|---|---|
| vm_cores | number of cores assigned to a VM |
| memory | RAM size assigned to a VM |
| cpu_speed | CPU frequency assigned to a VM |
| cpu_load | average system load of a VM over a 5-minute period |
| location | country code where the VM is located |
| availability | % of uptime of the whole provider infrastructure |
| stirage_location | country code where the given shared storage volume is located |
| storage_reliability | indicates the number of replicas for the given shared storage volume |
| reserve | number of VMs to be reserved for the given VirtualSystem |
| co_locate_rack | all the VMs from the given VirtualSystem must be allocated on the same rack |
| not_co_locate_host | all the VMs from the given VirtualSystem must be allocated on different hosts |
| minimum_LoA | minimum Level of Assurance that a user wants to be required for accessing her own resources |

Table 1: SLA terms in Contrail

Contrail provides autonomic support for the SLA life cycle, i.e., each provider is able to automatically negotiate its available resources and generate SLA offers completed with price. Automatic negotiation allows providers to personalize their offer, adapting it to the needs of each user or even of each single application: manual negotiation of users' SLAs would not be feasible for Cloud providers targeted to the consumer market. Automation is then

applied also to monitoring: when a provider is selected for provisioning it should be able to identify which attributes must be monitored and to automatically start monitoring for a specific user SLA; then, when the system is running, each SLA violation should be automatically handled; and finally termination of the SLA must be automatically managed when the agreement expires.

Another challenge is to find the right SLA syntax to connect each SLA term with the aspect of the service it should refer to. To describe IaaS services the OVF standard has been selected, therefore terms in the SLA must refer to items in the OVF, all with a backward compatible syntax.

Contrail adopts the SLA(T) (SLA Template) model proposed by the project SLA@SOI and extends it to use a standard OVF descriptor to specify virtual resources. OVF's VirtualSystems represent classes of Virtual Machines, OVF's SharedDisks represent external storage and all the elements can be connected in complex layered architectures through VLANs identified in the NetworkSections. To monitor / enforce SLA terms it must be known what the expressed guarantees are referring to and there should be a link between SLA terms (guarantees) and OVF items (resources). Contrail extends the SLA@SOI syntax to create such a link.

Generic SLAs, i.e., without an explicit connection to an OVF to avoid constraints on the amount of resources and therefore allowing scalability, have been evaluated then abandoned in favor of the possibility to associate more than one OVF to a single SLA. The OVF describes *types* of Virtual Machines but not their number, thus a main OVF is used for defining the initial application and an *incremental* one for scaling.

Scalability rules within the SLA itself, i.e., automatic scaling, can now be implemented by Java and Drooools actions specified in the SLA (Guaranteed Actions) that can ask for more resources when warning thresholds are violated (proactive SLA violation detection).

Elasticity support with SLAs poses the alternative between using the same SLA associated with multiple provisioning requests (the original one and any further requests to scale the system) and re-negotiating the SLA every time a need to scale arises. Contrail offers elasticity support by referring to the same SLA, because negotiation takes time and may need human decisions, whereas scalability needs to be fast to keep up with an increasing load.

The main outcome of Contrail is the definition and implementation of SLAs for federations of Clouds, which addresses several sub-challenges. Contrail defines a way to compare SLAs to negotiate with multiple providers and select the best one. A Federation might decide to split an application over multiple providers, thus the federation SLA Manager must be able to split the original SLA and then reconstruct it from the various pieces; splitting and aggregation operations have been defined (but not yet implemented). Other challenges for an efficient SLA Management at federation level are also SLA enforcement, monitoring, and accounting.

**Automatic negotiation of SLAs in Contrail.** To enable negotiation interoperability of different cloud providers with the Contrail federation, a SLA Management layer is added to each provider. This layer is able to understand the SLA syntax used by the federation and to automatically create SLA offers which will be proposed to the federation on behalf of the provider.

The model of interaction proposed by Contrail is based on multi-level negotiation of SLAs: a user negotiates a SLA with the Contrail federation, which in turn negotiates with several providers to select the best one that can satisfy the

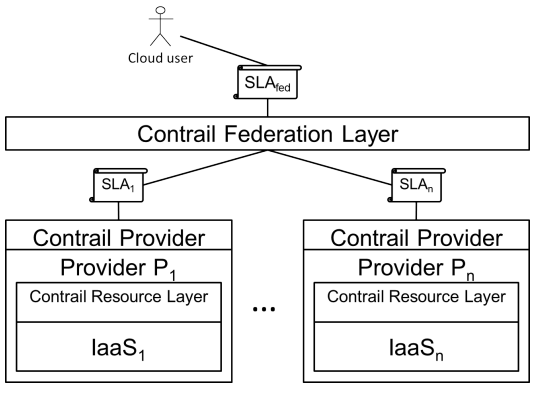


Figure 3: Multilevel SLA negotiation

expressed user needs (see Fig.3). In this case the Contrail system acts as a cloud broker.

The provider selection process has been implemented in a customizable way: the selection algorithm is defined as a plug-in and the Federation Administrator can select the most appropriate one. The default algorithm represents the SLAs offered by providers as points in a multi-dimensional space where each SLA term represents a dimension and the respective coordinate is proportional to the value of that term in the considered SLA. Direct proportionality is used for terms to be maximized, such as the amount of memory of a VM, and inverse proportionality is used for terms to be minimized, such as price. Federation SLA Manager compares the different providers' SLAs by calculating the euclidean distance of each SLA-point from the origin and selecting the farthest.

Users can influence the selection process by defining selection criteria that are applied as weights to the respective terms to privilege or penalize them. In this way for example the user can assign higher priority to providers offering faster VMs, in case of CPU-bound applications, or to providers offering more memory for memory-bound applications.

The final result of Contrail SLA Management has been a set of integrated components to enable SLA handling for Cloud Providers. Automated SLA handling is also the basis for the Federation of Clouds, which is Contrail's flagship functionality.

Commercial Cloud Providers currently don't offer automated SLA handling, thus the guarantees they offer cannot be easily personalized for the needs of each customer. Also the comparison between the offers of several different providers is not an easy task to be performed manually and automated SLA handling is a big enabler for it. SLA handling also includes monitoring functionalities which allow to automatically spot SLA violations and act consequently, either by increasing the amount of allocated resources (elasticity) or by moving the application elsewhere. The potential impact of an integrated Cloud stack with SLA handling and Federation functionalities enables small and local providers to compete in the Cloud market standing by the giants.

### 3.2.1 Monitoring support

The aim of Contrail is to support various providers and therefore various IaaS clouds in a unified federated manner. The Contrail monitoring service has been designed to add specific functionalities to OpenNebula and OpenStack, currently supported by Contrail. Contrail provides support for any IaaS solution by separating the logic for obtaining the metrics from the logic of storing and processing them. In addition, it implements conversion of a specific syntax of the metric for a given IaaS to the more general one as required by the Contrail system. The design of the solution is shown in the Figure 4.

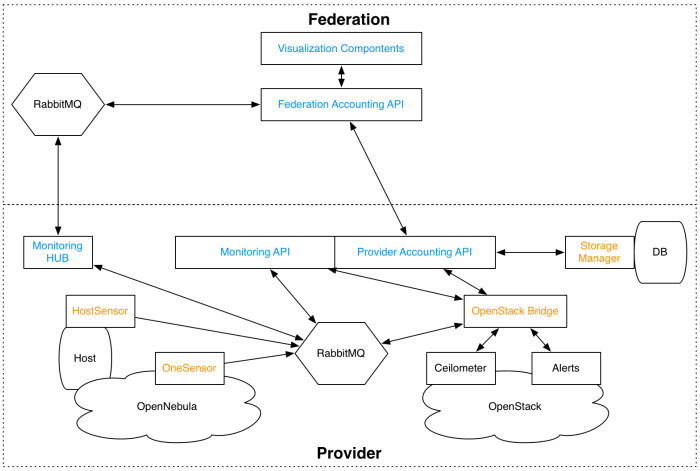


Figure 4: Monitoring architecture design for various IaaS support.

The metrics on the OpenNebula are gathered with the *OneSensor* module. This module implements RPC hooks and translates the plain text OpenNebula format of the results to a more detailed metric format in XML, which includes the source of the metric as well. This additional information is needed by the Contrail system to make properly mapping between user applications and the metric. Another low level component is

*HostSensor* which translates the host metrics to the same common format. This uniform description is then sent over the messaging system to the *Storage Manager* component that implements storage management policies to reduce the volume of information. The API on the provider side gives access to the metrics stored in the Storage manager. All events like notifications and violations can be pushed to the Contrail Federation over the Monitoring Hub which presents a secure communication channel between messaging systems on different networks.

Unlike OpenNebula, OpenStack already has a monitoring component Ceilometer, therefore instead of implementing hooks, Contrail implements a bridge that translates Contrail requests to OpenStack requests on the API. The mapping is between Contrail IDs like the user, the application, and Virtual Execution Platform (VEP) information, and between OpenStack IDs on Keystone and Nova. Besides the mapping of identities, the mapping also includes translation between formats of the metrics and between alarms and notifications syntaxes. The Contrail monitoring support has proven to be efficient and planned to be transferred into OpenStack, first as extensions of exiting BluePrints as well as a plan for adding new BluePrints for the next release of OpenStack.

### 3.3 Security

Contrail security solutions aim at achieving a secure infrastructure and delivering the level of isolation of users of multi-tenant systems, whilst not compromising usability – too much security and users will try to circumvent it and not use the system; too little and the system cannot be trusted. Contrail security provides a framework and infrastructure to support Federated Identity Management, Authentication and Authorisation services. These services allow fulfilling guarantees such that users of Contrail will be able to use the system without their personal information (user account details) being disclosed, and with the confidence that data they store in the system will not be liable to corruption or illegal access. Cloud providers can be assured that different Contrail applications are separated by network and data storage boundaries and cannot interfere with each other.

Security support is a critical component of the Contrail system, because potential customers of the Contrail framework (i.e., deployers of cloud federations) will adopt Contrail only if they feel confident in the security, and potential end users will use Contrail-based federations to run their cloud applications only if they feel the environment is secure. The Contrail framework must provide adequate and comprehensive security support, tailored for the needs of Cloud computing (specifically, it must be elastic, thus integrate security solutions to isolate elastic applications or verify whether a user is authorised to access additional resources at runtime.) In fact, since the Cloud environment is mainly targeted at business users, an enhanced level of security support is a key factor in allowing the adoption of Contrail-based cloud federations, because business users' data could have a high economic value and data theft or damage could result in considerable financial and reputational loss.

Security in Contrail includes authentication, authorisation, delegation, elastic security, and accounting – together known as "ConSec". The design of the Contrail security (ConSec) components has been performed while designing the rest of the Contrail framework, thus achieving a high level of integration with the other framework components.

In Contrail, the IaaS services run on the provider's infrastructure, and under the providerÕs terms and control. Therefore, guaranteeing protection at the infrastructure level concerns several issues: i) isolation of the component per se with protection from unauthorised users and Cloud providers; ii) protection and integrity of the data stored on Cloud IaaS services; iii) protection and integrity of the data exchanged between users and Cloud IaaS services. The internal security is not only achieved at the Contrail component level but it is also determined by the level of protection that can be achieved in IaaS providers and platforms.

Contrail defines a **"federated" identity** management scheme to allow users to use external identity providers (IdPs) – endpoints which are capable of, and trusted by the federation to, make assertions about the user. Very briefly, the sort of services expected from an external IdP can include: identity assertions about the user, authorisation attributes, traceability, and a certain level cryptographic strength and protocol design associated with the authentication process. Not all IdPs need provide all this information, but together they add up to a certain *level of assurance* (LoA) which is used in Contrail as a QoP parameter. The important advantage is that the infrastructure is able to assign a consistent LoA to each IdP, and to use this LoA as a basis for allocating services: this approach is essential to supporting the diversity of the use cases, where users already have existing identities from IdPs outside of Contrail, and communities with higher security requirements can choose to allow only those with the highest LoA. Contrail currently supports Shibboleth federations and OpenID. These identity providers might use different technologies, and publish attributes inconsistently, even when using the same schema. Contrail harmonises the attributes published, supplementing internal credentials with the relevant authentication and authorisation attributes when not available from external providers.

Contrail **authorisation** solution ensures that authorisation attributes are communicated securely to Providers to ensure that authorisation attributes are always available and enforce access control, against policies maintained at the federation level. Access control can be managed on *volatile* attributes, which might change during the lifetime of the users activity and require a re-evaluation of the user's permission to use the resource, and subsequent follow-up action such as termination or suspension of the user's activity. The Contrail authorization support is based on the UCON (Usage Control) model [27]. It introduces new features in the decision process w.r.t. traditional access control models, such as (i) *mutable attributes* of subjects and objects and, as a consequence, (ii) the *continuity of policy enforcement*. Mutable attributes describe features of subjects and objects that change due to the decision process, e.g., users' reputation and resources' workload. Mutable attributes lead to the need of continuously monitoring their values, and re-evaluating the security policy to guarantee that the right of a subject to use the resource holds while the access is in progress.

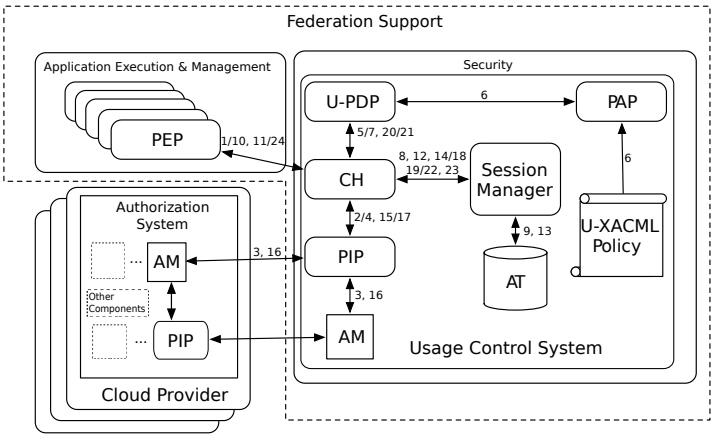


Figure 5: Integration of the UCON System within the Contrail Cloud Federation

This model can be successfully adopted in case of long-standing accesses because the decision process is performed continuously during the access time. The *pre-decision* phase corresponds to traditional access control, where the decision process is performed at the request time to produce the access decision. The *ongoing decision* phase, instead, is executed after the access is started and implements the continuity of control that is a specific feature of the UCON. Continuous control implies that policies are re-evaluated each time mutable attributes change their values. The UCON decision process evaluates *authorizations* (predicates over subject's and object's attributes), *conditions* (predicates over environmental or system status), and *obligations* (actions that must be performed along with the access). If the decision process detects a policy violation while an access is in progress, this access is revoked and resources are released.

Contrail exploits the U-XACML policy language [6, 19] to encode UCON policies. The U-XACML

extends the XACML language [23], the widely used access control language, with constructs for usage (continuous) control and attribute updates occurred as a result of the access. The Usage Control system architecture for the Cloud federation is shown in Figure 5. It extends the common authorization system architecture [23] to deal with a continuous policy enforcement.

Accounting ensures that accounting records are timely, accurate, and secure (*e.g.* against repudiation, alteration, or falsification.)

Contrail manages the internal delegation of federation credentials with OAuth. Contrail leverages the emerging OAuth2 protocol to delegate the rights to obtain credentials on behalf of users. The implementation has been extended to manage pre-authorisations and to allow the user to monitor the delegations. OAuth could also be used to manage the authentication, *e.g.* by having users authenticate with an external provider which uses OAuth.

The Contrail security solutions have proven to be successful from external projects reusing ConSec; particularly in the interests from EUDAT (eudat.eu) and ESGF (earthsystemgrid.org). For the former, ConSec is as of this writing part of the EUDAT AAI infrastructure. As regards the latter, ConSec delivered (in collaboration with ESGF) components which support internal certificates with OAuth delegation.

## 3.4 Contrail Provider services

The Contrail provider services manage the physical resources of a cloud provider. In Contrail, each cloud provider runs a copy of the Virtual Execution Platform (VEP) software which in turn seamlessly integrates its resources with the Contrail Federation. A key objective of the Contrail project is to provide a reliable cloud platform, i.e., users can trust the services offered by Contrail system. The gateway is the Contrail Federation and three other important components are VEP, GAFS, and VIN.

### 3.4.1 Virtual Execution Platform

VEP [3, 15, 18] is a cloud middleware software that interfaces IaaS cloud providers offering users a seamless way of using resources from multiple heterogeneous clouds. Cloud providers might have different means to manage VMs or networks, different image formats that can be deployed on a physical host, different interfaces, or different contextualization methods for the physical resources. VEP provides a uniform way of representing and managing the resources of a cloud provider.

VEP sits on top of IaaS management systems such as OpenNebula, OpenStack or any IaaS cloud providing an OCCI or EC2 interface. Users interact with VEP to deploy and manage their applications as a whole, while administrators use it to configure the IaaS cloud and limit the user access rights.

**Application Description** To deploy an application in VEP, the user specifies the application characteristics in a document [11] edited in OVF format, i.e., a format for the packaging and distribution of software to be run in VMs. This OVF format allows users to deploy applications on heterogeneous clouds as it is not tied to any particular hypervisor or processor architecture. The user can use the OVF document to communicate to VEP the following information:

- References to external files necessary to build the application, e.g., disk images. These references can be a filesystem path or a URL;
- VM disk images together with their image format;
- Shared storage to be mounted in the VMs;
- Virtual networks to be used by the VMs;

- VM templates defining the virtual hardware of the VMs, the VM disk images and the networks connected to the VMs;
- Virtual hardware specification,e.g., number of processors and cores, memory size, disks, network interfaces, etc. To improve portability, the format also supports alternative hardware descriptions, allowing the IaaS provider to select the configuration it supports, e.g., the disk image format;
- Contextualization data to be passed to the VMs at boot time, e.g., public key, IP, etc.

Using the OVF document, VEP deploys the user's application in a virtual execution environment, i.e., a generic platform composed of virtual resources, e.g., VMs, network or storage. However, to be deployed on the infrastructure, this virtual execution environment needs to be mapped to a set of virtual resources provided by the IaaS management platform. These resources are specified in VEP through virtual resources handlers, which basically define different classes of resource configurations, by following the DMTF's CIMI [12] standard[2]. Currently, VEP supports three types of handlers: amount of computational resources assigned to a VM; specific network setup; and type of storage for the application's data.

The Contrail federation and the user negotiate the SLA associated to an application, which is then deployed by VEP respecting those negotiated constraints [18]. The SLA support during the deployment is achieved with the definition of the Contrail Constrained Execution Environments (CEEs), which can be derived from a negotiated SLA or made available as templates ready to be instantiated by users. The CEE on which a new application is to be deployed can be specified by the user, the federation acting on behalf of the user, or from default rules. It is possible to deploy multiple applications on the same CEE, for instance applications sharing the same virtual network or storage.

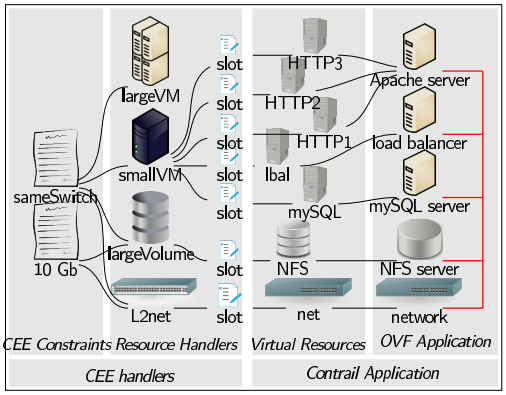



Figure 6: Constrained Execution Environments and OVF mapping

A CEE defines a virtual infrastructure made of resource handlers and constraints where user applications are deployed and managed. Fig. 6 shows the mapping between the resources described in the application OVF document and the CEE resource handlers specifying the constraints which should be respected for the deployment of each resource. Each resource handler specifies the physical resources to be allocated for each virtual resource (virtual machine, storage, or network) instantiated in the infrastructure. Different types of constraints are supported in VEP concerning performance, security, placement or the number of virtual resources which can be allocated. For instance, constraints can specify relations between resources, such as affinity to allocate resources close to each other in order to improve interactions, or anti-affinity to increase dependability, for instance to place virtual machines on different data centers. The CEE also defines the monitoring configurations, which are then used by the provider and the federation to evaluate whether a SLA is enforced. New deployment requests for additional resources of an application can then be automatically generated by the SLA enforcement services in reaction to performance indicator deviations, or directly requested by the user. Adding new resources to the application does not necessitate any SLA re-negotiation as long as the CEE constraints are respected.

As part of the SLA support, VEP can also facilitate the elasticity of applications, i.e., by adding or removing some of their resources, as required to meet their performance objectives. The elasticity support is facilitated by VEP through supporting two deployment modes of OVF applications: implicit and controlled.

[2]Compatibility tests with the standard are currently under evaluation.

In an implicit mode, all OVF virtual resources are deployed and started; in a controlled mode a user can explicitly specify which and how many OVF virtual resources listed in a *deployment document* should be started. By submitting a new deployment document to an existing application, the user can add more resources during the application runtime.

Other features of the Contrail VEP not extensively discussed in this section are: partial application deployment to allow a user to link the deployed application with an existing one, and design support for application snapshots to improve dependability for long running applications. VEP also handles advance reservation of resources before application deployment through its *Reservations Manager* service; reservations are stored in a database so that VMs placement can be done according to reservation requests, resource availability and user-defined constraints. This guarantees resource provisioning in the future.

#### Supported IaaS

The currrent release of VEP supports OpenNebula and OpenStack IaaS managers. Although these managers don't implement the same contextualization means (cdrom for OpenNebula and meta-data server for OpenStack), VEP allows to deploy the same system images on both systems through a dedicated contextualization script. Support for extra IaaS managers models is planned in the future through OCCI [24].

### 3.4.2 GAFS – the cloud file system

GAFS [17, 33] is a reliable and distributed file system. It splits the file-contents from the metadata and stores it in different services. The file content is stored in object storage devices (OSD) and the metadata is stored in metadata and replica catalogs (MRC). Separating them allows to easily scale the system and use different kinds of hardware for the different services. It provides a light-weight mechanism for creating volumes. Volumes are similar to generic file system as they provide their own namespace. Instead of creating a `/home` file system shared by all users, in GAFS it is easy to have one volume per user. The client is available for Linux, Mac OS X and Windows.

In Contrail, it serves as a general purpose Storage-as-a-Service file system that can be used from within the cloud (i.e. the user's application) but also from the outside via the Internet. GAFS is unique in Contrail because it is used at two levels in the software stack. At the IaaS layer, it provides the storage necessary for operating the Contrail infrastructure, like storing the customer's VM images or system logs. At the PaaS layer, it can be used via ConPaaS to provide storage to applications.

Cloud storage is generally expected to behave like an exclusively owned, highly available, reliable and responsive storage resource that has unlimited capacity and offers a high degree of data reliability and security. From a provider's perspective, it has to be maintainable with a minimal effort and provide a high degree of flexibility in order to serve the needs of different users. To fulfill these requirements, various non-standard features are implemented in GAFS: replication with automatic and transparent replica failover, self-tuning, self-healing and enhanced monitoring capabilities, quality of service, as well as different interfaces, like e.g. POSIX and HDFS.

**ConPaaS.** The XtreemFS service in ConPaaS (see Section 3.5 for a detailed description of ConPaaS) can be considered as one of the best integrated ones. It implements the consistent addition and removal of replicas which allows to scale the service depending on the user's demand. We also provide support for SSL encrypted connections and persistent storage. Now, it's possible to suspend a complete XtreemFS service in ConPaaS and resume it on different VMs without losing data. Additionally, we improved the HDFS adapter and conducted benchmarks which showed that GAFS is as fast as the stock HDFS implementation. This

allows the MapReduce service in ConPaaS to use GAFS for input and output data. By storing the data in GAFS instead of the Hadoop file system, it is easier to integrate MapReduce with legacy applications.

**Hosting for VM images.** Important results have been achieved for scalable deduplication and scalable multi-deployment of VM images. By using GAFS' two replication methods for immutable and mutable data, we implemented a scalable and highly-available storage system for virtual machine images. The original image is considered immutable and new replicas can be considered as a cache of the original file. When a new VM is started, a local replica of the original image is created on-demand on the node hosting the VM. All writes to the image are directed to a second file which is replicated with read-write replication. This method essentially implements an efficient copy-on-write mechanism for VMs exploiting GAFS unique features. The evaluation shows that the system scales with the number of VMs while less than 4% additional disk space was used by additional replicas per VM.

**Releases.** XtreemFS 1.5 has been released in March 2014 and provides packages for most major Linux distributions using the Open Build Service (OBS).

### 3.4.3 Contrail Virtual Infrastructure Network

The Virtual Infrastructure Network (VIN) provides connectivity among all resources of a Contrail application, consisting of virtual compute nodes, external GAFS storage servers, and physical client machines. An important property of Contrail applications is that they are elastic, able to add or remove resources at runtime. Such elastic applications are not supported directly by cloud infrastructure stacks such as OpenNebula and OpenStack in terms of network connectivity. VIN provides network tunnels among the entities of a Contrail application. Along with dynamic scaling (adding and removing) of nodes, VIN adds and removes the according tunnels on the fly. Tunnels can be encrypted using IPsec or unencrypted using GRE.

A Contrail application generally consists of multiple VMs, and these VMs generally need to communicate with each other, with the public Internet, and with the user, and with external GAFS server machines. VIN implements different implementation flavours that support each of these types of machines. Together, they can be used to integrate federated cloud resources with VIN overlay networks.

VIN instances are dynamically created as part of starting up Contrail applications. A VIN instance is created via the Contrail federation service. The federation triggers the startup of the corresponding VIN (via VEP). Co-located with the federation service, a central, dedicated VIN controller is executed on the application's behalf. On each cloud node (running the virtual machines belonging to an application), so-called VIN agents are running that establish the necessary connectivity. VIN agents are managed by the central VIN controller.

Two versions of VIN agents are available. In order to support virtual machines that are unaware of the fact that their network connection is provided via a virtualized VIN overlay, one implementation of the VIN agent operates completely on the physical host that supports the virtual machine. This implementation is applicable if the Contrail infrastructure (VEP) is starting up the application, requiring the virtual machine to be configured for supporting hot plugging of network interfaces.

The second implementation of VIN agents is running inside the virtual machine itself. This implementation is for integrating public cloud infrastructures, that are not running Contrail infrastructure services. Here, the virtual machine image itself needs to install the VIN agent; this version is only available for virtual machines running the Linux operating system.

The "VIN agent" actually consists of several components:

- the VIN agent itself, communicating with the application's central VIN controller, and further managing VIN operation on the host;

- the *strongSwan* [30] IPsec implementation for Linux that allows configuring the kernel's IPsec connections according to VIN's needs;
- the *styx* plugin for *strongSwan* that allows dynamic reconfiguration of IPsec connections;
- the *safeconfig* environment that allows execution of configuration scripts under restrictive control of the host administrators.

Besides (virtual) compute nodes, Contrail applications need to integrate GAFS servers in order to perform useful computations. For achieving both connectivity and protection of these file server accesses, VIN allows to integrate GAFS servers into an application's VIN. Here, the VIN agent (and its accompanying packages *strongSwan*, *styx*, and *safeconfig*) run on the server host. Communication between the VIN agent and the application's central VIN controller is identical to the case of supporting virtual machines. Hence, these server nodes are seamlessly integrated into a Contrail application.

The integration of client machines (outside a cloud environment) is achieved using the same implementation as intended for GAFS servers. For both types of machines, the requirement of our implementation is that they run the Linux operating system.

The integration of VIN and Contrail's security services is achieved via VEP. VEP is starting up VIN agents, by using contextualization mechanisms of the underlying cloud infrastructure stack. By doing so, it provides them with the necessary security credentials for the VIN Certification Authority (CA). For example, with OpenNebula, VEP is using VM templates to trigger the startup of the VIN agent and to hand it the necessary credentials as template parameters.

### 3.5 Contrail's Platform as a Service (ConPaaS)

ConPaaS [26] is the platform as a service component of Contrail. ConPaaS is an open source runtime environment for hosting applications in the cloud which aims at offering the full power of the cloud to application developers while shielding them from the associated complexity of the cloud.

ConPaaS is designed to host both high-performance, scientific applications and online Web applications. It runs on a variety of public and private clouds, and is easily extensible. ConPaaS automates the entire life cycle of an application, including collaborative development, deployment, performance monitoring, and automatic scaling. This allows developers to focus their attention on application-specific concerns rather than on cloud-specific details.

ConPaas is being deployed via a management console, providing both a Web-based frontend and a programmatic interface. Via this console, ConPaaS applications can be managed that consist of one or more individual ConPaaS services. Via the site `www.conpaas.eu`, a publically available testbed installation can be accessed.

The ConPaaS runtime is organized as a collection of services, where each service acts as a replacement for a commonly used runtime environment. For example, to replace a MySQL database, ConPaaS provides a cloud-based MySQL service which acts as a high-level database abstraction. The service uses real MySQL databases internally, and therefore makes it easy to port a cloud application to ConPaaS. Unlike a regular centralized database, however, it is self-managed and fully elastic: one can dynamically increase or decrease its processing capacity by requesting it to reconfigure itself with a different number of virtual machines. In the following, we describe the services currently included in the ConPaaS platform, and the underlying infrastructure providers that are supported.

### 3.5.1 Scalarix

Scalarix [28, 29] is a distributed transactional key/value store. Instead of supporting arbirtrarily complex data schemes, like e.g. SQL databases, Scalarix only supports a single table with two columns, keys and values, with an index on the first column. All data accesses are wrapped in transactions. This restricted data scheme makes it easier to build a scalable database like Scalarix. The integrated routing mechanism allows low-latency data access even in deployments with thousands of nodes.

New functionalities have been recently implemented in Scalarix, e.g. improving the underlying algorithms for ring maintenance and transactions, a new performance monitoring system and an auto-scale-component. Scalarix belongs to the first set of services integrated with ConPaaS. The Scalarix auto-scaling mechanism integrated in ConPaaS provides a new monitoring system, which can quantify the current load and the distribution of the response latencies. Previously, the auto-scaler relied on simple averages. The new system allows tracking arbitrary quantiles.

### 3.5.2 MySQL

While Scalarix serves as the key-value store in ConPaaS, the MySQL service provides a SQL database. It supports full master/master replication with scale-out based on Galera Cluster for MySQL [14]. As an example of the use of MySQL service, WordPress, a PHP-based blogging software, is often used as a ConPaaS demonstrator. It serves as a typical for web-applications. It employs the PHP, the MySQL and the GAFS service. PHP for hosting the application code, MySQL for hosting structured data and GAFS for static content, like e.g. pictures.

### 3.5.3 MapReduce

Google's MapReduce [9] became the de-facto runtime environment for large-scale data-analytics. The most widely used Open-Source implementation of the MapReduce paradigm is Apache Hadoop. ConPaaS' MapReduce service is based Cloudera's Hadoop. Cloudera bases its distribution on Apache Hadoop but adds performance patches and further components, e.g. Cloudera also provides Hue, a browser-based GUI. That makes it especially easy for new users to start with MapReduce.

In ConPaaS, Hadoop's management service run in the first VM. When further VMs are added, they only serve as workers and increase the capacity of the Hadoop cluster. To make data-management easier, we developed an HDFS adapter for GAFS. Therefore, the input and output of MapReduce jobs can be stored in GAFS.

### 3.5.4 Task Farming

The ConPaaS TaskFarming service provides a bag-of-tasks scheduler [25] for ConPaaS. The user needs to provide a list of independent tasks to be executed on the cloud and a file system location where the tasks can read input data and/or write output data to it. The service first enters a sampling phase, where its agents sample the runtime of the given tasks on different cloud instances. The service then, based on the sampled runtimes, provides the user with a list of schedules. Schedules are presented in a graph and the user can choose between cost/makespan of different schedules for the given set of tasks. After the choice is made, the service enters the execution phase and completes the execution of the rest of the tasks according to the user's choice.

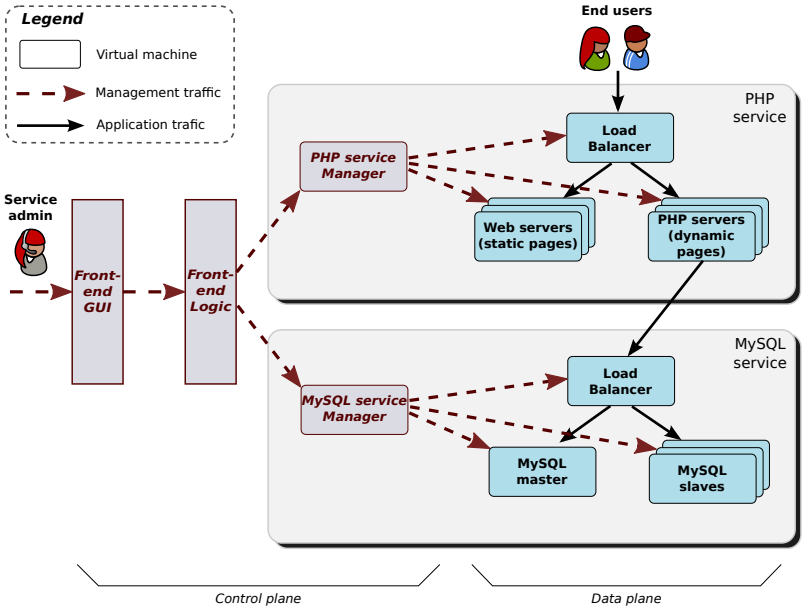



Figure 7: ConPaaS deployment of a Web hosting application.

The TaskFarming service uses GAFS for data input/output. The actual task code can also reside in a GAFS volume. The user can optionally provide a GAFS location for his or her service, which is then mounted on the virtual machines of the TaskFarming service.

### 3.5.5 Scalable Web Application Hosting

ConPaaS provides two dedicated Web hosting services. The PHP Web hosting service is dedicated to hosting Web applications written in PHP. Likewise, the Java Web hosting service is dedicated to hosting Web applications written in Java using JSP or servlets. Both services can also host static Web content. In a typical Web hosting application, either service is deployed in combination with a storage service, typically a MySQL database service. Figure 7 shows such a deployment of a PHP service along with a MySQL service.

ConPaaS has an autoscaling component for its Web application hosting services [13]. The ConPaaS autoscaling system scales a web application in response to changes in throughput (response time to the user) at fixed intervals. The autoscaler is built according to the architecture shown in Figure 8. There, the scaler component is in charge of controlling the other components. The dynamic load balancer is distributing incoming requests across all cloud resources available to the application at a given moment in time. The profiler measures in advance the operation performance of a given cloud instance type, which is used by the predictor that extrapolates the current web request traffic into the near future and takes decisions about scaling out or scaling back resources.

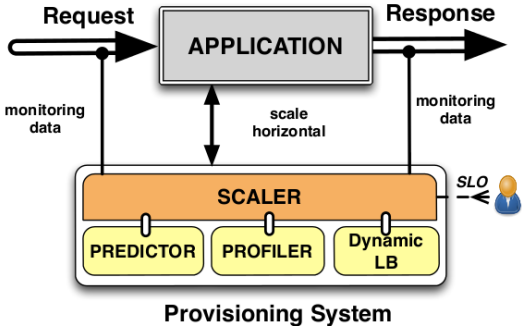



Figure 8: The ConPaaS Web-hosting autoscaling system.

The autoscaling systems provides three levels of scaling plans, referred to as gold, silver, and bronze.

A gold scaling plan provides the best possible service by selecting a higher cost strategy (using some level of overprovisioning). A silver scaling plan uses a median cost strategy, and a bronze scaling plan hardly selects any overprovisioning by favouring the strategy with the lowest cost. This kind of auto scaling aims at fulfilling a given, user-defined service-level objective (SLO), while letting the user express his or her priorities for trading cost vs. SLO compliance.

### 3.5.6 IaaS Infrastructures supported by ConPaaS

All ConPaaS service instances are implemented using a service core written in Python. This service core implements all the functionalities that are not specific to a particular service. Examples are starting and stopping service instances, instance health monitoring, and securing the communication among the service instances belonging to a ConPaaS application. Another important functionality of the ConPaaS service core is interfacing to underlying cloud infrastructures. This is achieved by using Apache Libcloud [21]. Currently, ConPaaS can access the following cloud infrastructure providers via Libcloud drivers: Amazon EC2, OpenNebula, OpenStack, Windows Azure, Contrail VEP, and Contrail Federation.

## 3.6 Contrail powered applications

Since the Contrail system is a base software infrastructure, it is difficult to show up directly its own potential. The Contrail Use Cases (UCs) are the reference applications that have been developed and deployed on the Contrail framework, in order to evaluate, test, and demonstrate the full set of technological features and functionalities developed in the Contrail components. The selection of the UCs has been driven by several scientific, technical and business considerations: a number of factors have been taken into account, such as heterogeneity of market scenarios, complementarities of the sectors addressed, and relevance both at industrial and scientific level.

The Use Case – Multimedia Marketplace – has been further expanded and transformed into a large scale and market oriented "Demonstrator", fully open to final users `http://contrail-project.eu/mps`. In that real production scenario, almost all critical aspects of the technology developed by Contrail have been tested under heavy load conditions, and information on performance, scalability, security, and usability issues collected and reported so that corrective actions and fine tuning of the Contrail framework can be accomplished.

### 3.6.1 Multimedia Marketplace

The Multimedia Marketplace is a web portal used by end users to enjoy media contents. The system provides some basic functionalities to let users to signup, subscribe to download and streaming services, and search videos. It also comes with a couple of advanced features for searching and recommending contents. For the advanced content search, a video-based face recognition system allows users to search videos containing an input person, that can be submitted to marketplace also by dragging and dropping a photo of him/her. As regards the contents recommendation, users are recommended a set of videos, found by analyzing the rating the other users gave to the contents seen.

Over the end users, there are other actors in this use case: the Content Providers supplying raw contents to the Marketplace, and the Technology Providers offering specialized services to transcode contents and adapt them for the specific users' devices. They both profit from joining the Federation. The Content Providers are interested in storage resources, because their business come from the contents they manage, and may want to join the Federation also for trading storage capabilities. In contrast, Technology Providers

sell low level services to external operators and are interested in selling their exceeding computational power in some periods.

Internally, the Marketplace is a standard model-view-controller (MVC) web application based on Joomla. The bottom layer includes a Data Access Objects (DAO) module connecting a MySQL server to store Joomla entities, a media module interfacing the advanced search component and a content recommendation module interfacing the data mining system. The middle and bottom layers also interface Content and Technology providers. The Joomla application and related database are deployed on ConPaas, using ConPaas web hosting service and the ConPaas MySQL service, respectively. The advanced content search, exploiting HP Autonomy ImageServer for face recognition facilities, needs to be deployed on VEP for its computational requirements. The Marketplace administrator must first train the ImageServer with a set of faces. When the Content Provider uploads new contents on GAFS, a custom internal component (VideoRecognizer) connected to the ImageServer scans each video in the background, for a sample set of frames, in order to look for known faces. Then, for recognized persons, another module adds new tags to the Marketplace database, so that users' search for that person will list related movies in the result.

The data mining system, used for content recommendation, is based on Apache Mahout and Sqoop. Users' similarity is obtained by recommendation algorithms implemented in Mahout, which can run on top of Hadoop in case of big volumes of input data. Last users' history is imported from the database to HDFS via Sqoop and similarly the recommendation results are exported back to the database. The Sqoop and Mahout jobs are invoked from a client machine running on VEP, whereas the Hadoop cluster is deployed using ConPaas map-reduce as a service.

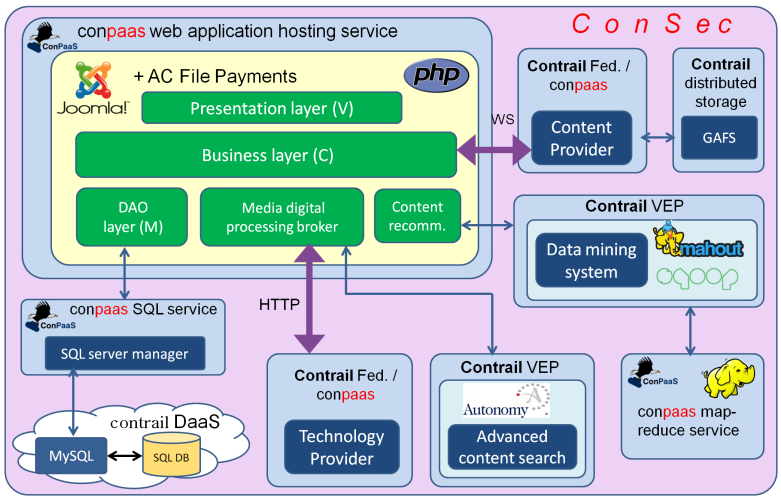


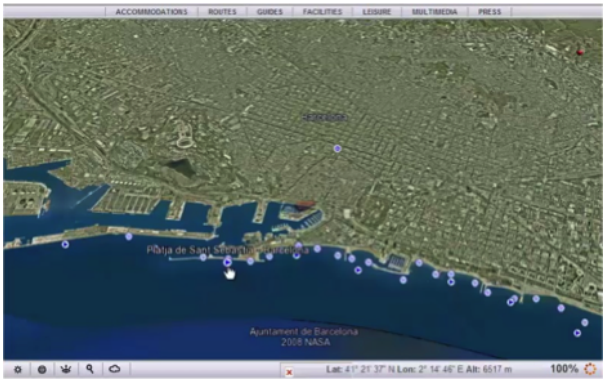

Figure 9: Multimedia Marketplace

The main requirements of this use case come from the Marketplace owner or administrator, who needs to put the Marketplace application on the cloud in order to benefit the speed and flexibility of the deployment, and control the elasticity of the system. Then there are other requirements that are far more difficult to be satisfied in a typical cloud: the privacy of users' data, the security of the infrastructure, the legal and geographical constraints on storage, the performance of most demanding instances in terms of CPU (face recognition), and reliability, both for storage and VM instances. To guarantee these key aspects to be satisfied, they are expressed inside an SLA that the owner negotiates with the Federation. The following SLA terms are needed for this use case: CPU speed, number of cores, memory, minimumLoA, VM location, storage location, storage reliability and provider availability.

### 3.6.2 Distributed provision of geo-referenced data

The Distributed Provision of Geo-referenced Data implements a 3D Virtual Tourist Guide (VTG) through Web access to interactive digital maps and geo-referenced multimedia content. Users can zoom maps on a specific area of the globe and visualize them at different levels of detail, depending on available detailed information about the region. Layers can cover the territory partially or completely.

Added value is given by Points Of Interest (POIs) related to tourism (like hotels, restaurants, museums), historical information about monuments or places, weather forecasts, images shared by users, etc.




**Map planning and POIs representation**


**Point of Interest and correlated image/videos**

The Virtual Tourist Guide service relies on the concept of Federation offered by the Contrail framework. It enables an open and distributed platform where companies and institutions, that are willing to share their maps or geo-referenced content, are allowed to add new terrain layers and POIs to VTG infrastructure. This will greatly enrich the quality of experience of the final users.

The service is managed by an Application Provider that monitors the quality of the overall service. Several Data Providers (including both Content Providers and Spatial Data Infrastructures) can join the service federating their private or public IT infrastructure: still maintaining full control over their data they can provide geo-spatial data and geo-referenced content from different sources to a single viewing environment.

Main features of VTG are:

- User-friendly interface for an immersive navigation experience;
- Search and fly to geographic location;
- Dynamic activation of POIs.

Data Providers are encouraged to contribute their services, since the Contrail technology guarantees a high level of security, protection, and reliability in data storage and management. The VTG platform could lower the initial investment needed for Data Providers: they could rent IT infrastructure from a Cloud provider in the Contrail Federation or they could easily federate their own private/public Cloud with Contrail. Quality of Service (QoS) is guaranteed by transparent scalability and elasticity mechanisms provided by Contrail Federation. All this at reasonable costs. The following benefits have been perceived by using Contrail:

- Dynamic platform with open APIs that can be easily joined by data and content providers;
- Locality of data assured: content providers can tell where their data may be used and by which user categories, thus keeping full control on them;
- Protection of data is guaranteed not only by Quality of Protection (QoP) rules, but also all data channels are encrypted to preserve confidentiality;
- Resiliency and distribution of Geodata and POI data can be easily managed and enhanced with the use of GAFS features;
- Interoperation of Contrail clouds with OpenStack based clouds; in this way providers that already manage OpenStack infrastructures can easily join the Contrail federation.

### 3.6.3 Scientific Data Analysis

The Scientific Data Analysis (SDA) web portal allows scientific end-users to automatically classify the shapes of chemical samples in neutron-scattering experiments using the ISIS facility at STFC. The challenge the SDA portal addresses is matching experimental data to physical shapes. Instead of the end-user having to guess at various parameters, running a simulation and then looking at the closeness of the match between the simulation output and the experimental data, the portal runs hundreds or thousands of simulations and presents the best matches to the user.

The portal allows users to upload data and define a parameter sweep on relevant model variables. A library of shape models is available for the user to select; it contains candidates to match the experimental data. Other variables related to the shape models (such as the ranges of physical dimensions to consider) can then be specified. The portal sets the simulations running on the Cloud. Application logic in the portal compares the simulation results with the experimental data and presents the user with a display of which shapes are the closest matches to the experimental data.

The portal takes the values defined in the parameter sweep and constructs a Bag-of-Tasks (BoT) file for the ConPaaS TaskFarm. Each line of the BoT file defines an invocation of the shape modelling code with a distinct combination of the parameter values. The portal monitors execution of the Bag-of-Tasks, showing the user a progress bar marking percentage completion of the tasks running in the TaskFarm. The completed simulation results are copied back from GAFS to the server hosting the portal, where an analysis routine selects the shape models which are the best fit to the experimental data. The portal then produces graphs showing the best matches between the simulation and the experimental data.

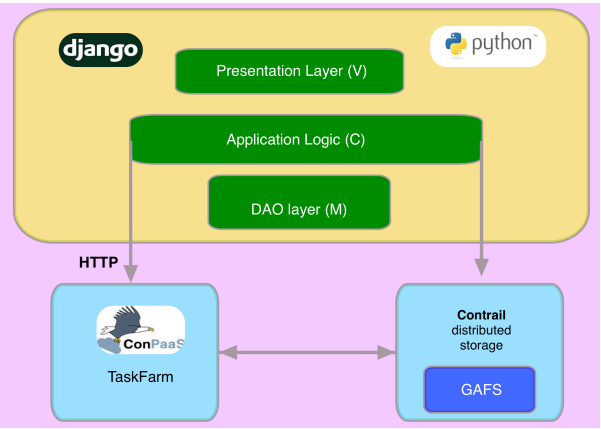



Figure 10: Scientific Data Analysis

Contrail features, ConPaaS TaskFarm and the GAFS distributed storage, proved to be important for this use case. The TaskFarm allows a number of independent shape modelling simulations to be run in parallel. The GAFS distributed storage is used by the SDA portal to upload experimental data to the ConPaaS TaskFarm and to download the simulation results, without the user needing to manage the data.

## 3.7 Building on top of Contrail

The Contrail project provides an open-source integrated approach to virtualization at the IaaS and PaaS levels [1]. This document discussed the challenges and the approach in Contrail to address interoperability issues and SLA-aware deployment of distributed applications across a federation of heterogeneous cloud providers. We described the components of the Contrail architecture and we outlined the key features and the design of the Contrail services.

Contrail allows dynamic transparent leasing of resources from multiple sources and ease the user access to cloud services. Moreover, Contrail offers a set of elastic PaaS services with multi cloud support. In a nutshell, Contrail improves the effectiveness of the pay-as-you go approach and increases the end-user freedom in the provider selection. Moreover, strong SLA guarantees support both provider competition (on prices) and collaboration (provider aggregation) to access new market segments.

The impact of the Contrail project toward the defragmentation of the market and future cloud is considerable. In fact, many collaborative projects and organisations are adopting or investigating the results of the Contrail project.

The Contrail topics remain highly relevant in H2020 and EIT ICT Labs future cloud action line. It worth mentioning that Contrail paves the way to achieve high performance heterogeneous clouds with dynamic and automated provisioning and orchestration of cloud resources. The usage of standards and the Contrail solutions for secure and trusted cloud platforms with federated identity and SLA support are a first step to address the European cloud market needs.

# References


[1] Contrail project. `http://contrail-project.eu/`. Code available at `http://contrail.projects.ow2.org/xwiki/bin/view/Main/WebHome`.

[2] Etsi Cloud Standards Coordination. `http://csc.etsi.org/website/home.aspx`.

[3] VEP: Virtual Execution Platform. `https://project.inria.fr/vep/`.

[4] R. G. Cascella, L. Blasi, Y. Jégou, M. Coppola, and C. Morin. *Contrail: Distributed Application Deployment under SLA in Federated Heterogeneous Clouds*, volume 7858 of *Lecture Notes in Computer Science*, pages 91–103. Springer, 2013.

[5] Cloud Data Management Interface (CDMI) Version 1.0.2, June 2012. `http://www.snia.org/cdmi`.

[6] M. Colombo, A. Lazouski, F. Martinelli, and P. Mori. A proposal on enhancing XACML with continuous usage control features. In *CoreGRID ERCIM Working Group Workshop on Grids, P2P and Services Computing*, pages 133–146. Springer US, 2010.

[7] CompaptibleOne. `http://www.compatibleone.org/`.

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

[9] J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. *Communications of the ACM*, pages 1–13, 2008.

[10] DeltaCloud. `http://deltacloud.apache.org/`.

[11] DMTF. Open Virtualization Format Specification. `http://www.dmtf.org/standards/ovf`, 2010.

[12] DMTF. Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol, An Interface for Managing Cloud Infrastructure (v1.0.1 DSP0263). `http://www.dmtf.org/standards/cloud`, 2012.

[13] H. Fernandez, G. Pierre, and T. Kielmann. Autoscaling Web Applications in Heterogeneous Cloud Infrastructures. In *IC2E 2014, IEEE International Conference on Cloud Engineering*, Boston, MA, USA, March 2014.

[14] Galera Cluster for MySQL. http://codership.com/.

[15] P. Harsh, F. Dudouet, R.G. Cascella, Y. Jegou, and C. Morin. Using open standards for interoperability issues, solutions, and challenges facing cloud computing. In *8th International Conference on Network and Service Management (CNSM)*, 2012.

[16] IETF OASIS WG. RFC6749 – The OAuth 2.0 Authorization Framework, 2012.

[17] Michael Berlin Jan Stender and Alexander Reinefeld. Xtreemfs: A file system for the cloud. In *Data Intensive Storage Services for Cloud Environments*, pages 267–285. IGI Global, 2013.

[18] Y. Jegou, P. Harsh, R.G. Cascella, F. Dudouet, and C. Morin. Managing OVF applications under SLA constraints on contrail virtual execution platform. In *8th International Conference on Network and Service Management (CNSM)*, 2012.

[19] A. Lazouski, G. Mancini, F. Martinelli, and P. Mori. Usage control in cloud systems. In *7th International Conference for Internet Technology and Secured Transactions (ICITST-2012)*, 2012.

[20] A. Lazouski, F. Martinelli, and P. Mori. A prototype for enforcing usage control policies based on XACML. In *Trust, Privacy and Security in Digital Business*, number 7449 in LLNCS. Springer, 2012.

[21] Apache Libcloud. http://libcloud.apache.org/.

[22] Peter Mell and Tim Grace. The NIST definition of Cloud Computing, version 15. http://www.nist.gov/itl/cloud.

[23] OASIS. extensible access control markup language (xacml) version 3.0. Technical report, OASIS Standard, 2010. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf.

[24] OGF Consortium OCCI WG, 2013. http://occi-wg.org/.

[25] A.-M. Oprescu and T. Kielmann. Bag-of-Tasks Scheduling under Budget Constraints. In *Proc. 2nd IEEE International Conference on cloud Computing Technology and Science (CloudCom 2010)*, 2010.

[26] Guillaume Pierre and Corina Stratan. ConPaaS: a platform for hosting elastic cloud applications. *IEEE Internet Computing*, 16(5):88–92, 2012.

[27] R. Sandhu and J. Park. The $\text{UCON}_{\text{ABC}}$ usage control model. *ACM Transactions on Information and System Security (TISSEC)*, 7(1):128–174, 2004.

[28] Scalaris. https://code.google.com/p/scalaris/.

[29] Thorsten Schütt, Florian Schintke, and Alexander Reinefeld. Scalaris: Reliable transactional p2p key/value store. In *Proceedings of the 7th ACM SIGPLAN Workshop on ERLANG*, pages 41–48, Victoria, BC, Canada, 2008.

[30] StrongSwan, the OpenSource IPsec-based VPN Solution. http://www.strongswan.org/.

[31] S. Sundareswaran, A. Squicciarini, and D. Lin. A brokerage-based approach for cloud service selection. In *IEEE Fifth International Conference on Cloud Computing (CLOUD)*, 2012.

[32] Nick Trigg. Standardization Report. Deliverable D16.5, CONTRAIL Project, April 2013.

[33] XtreemFS. http://www.xtreemfs.org/.

# 4 Potential impact

## 4.1 Dissemination activities

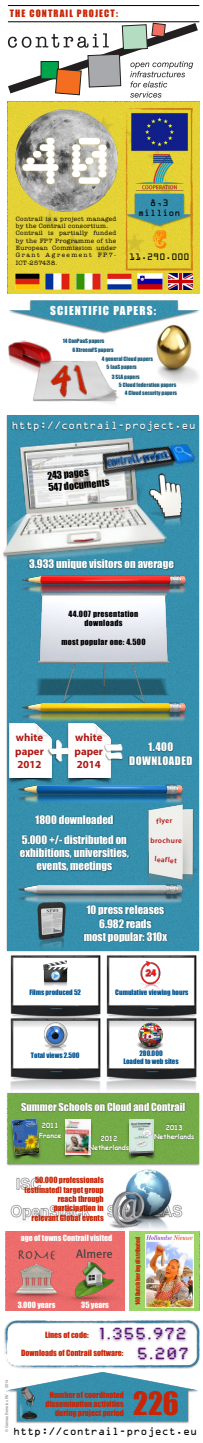



Figure 11: Infographic

The Contrail achievements are highlighted through a variety of dissemination tools, including scientific papers, attendance at major conferences to present the project results, the organisation of three Summerchools during the whole project period at which the students have been trained in using and applying the Contrail software.

Dissemination also included the distribution of several issues of Contrail component flyers. The most recent series of flyers contains materials on ConPaaS, XtreemFS, VEP, Contrail Security, the Demonstrator and a flyer targeted to business audiences. The Contrail consortium also issued a new version of the White Paper in January 2014. This document is a wholly revised and expanded version of the first publication in November 2012.

A series of videos have been produced where Contrail partners explain the relevance and use of the Contrail software to a diversified audience. These videos have been produced at conferences, Contrail co-organized events, Contrail meetings and workshops and have been posted on the Contrail website. The Contrail portal has been updated during the whole project period with news blogs, software updates, slides of presentations at different events, videos, and Contrail activities.

The infographic in Figure 11 summaries the project results. In the remaining of this section we cover the major events.

**Contrail Cloud Computing Business Day.** For a business audience a specific event was organised close to the end of the project to be able to include the latest Contrail developments and present them to business managers and public organizations. On January 23, 2014, the Contrail project organized a the Cloud Computing Business Day in Rome.

The Contrail Business Day addressed the following topics: avoiding vendor lock-in, combining private and public clouds, creating a single sign on and identity management. These are important features to enrich the offer and to smash the perceived barrier to more widespread cloud computing. SLA support and application management in federated Clouds are the key to open new opportunities for companies' usage of the Cloud.

The business day featured an in-depth overview of the current state of Cloud computing. In addition, a number of results, achieved in the European Contrail project, were presented and demonstrated that address some of the Cloud computing issues faced by Cloud users.

**ConPaaS workshop.** On June 13, 2013, the First ConPaaS Workshop was organized at the Vrije Universiteit in Amsterdam in collaboration with the Harness and MC-DATA projects (http://www.conpaas.eu/1st-conpaas-workshop/). Contrail speakers from VU, XLAB and ZIB presented the ConPaaS software in all its aspects. All presentations have been videotaped and available at:

- http://vimeo.com/contrailproject/videos/search:ConPaaS/sort:date;

- http://www.conpaas.eu/videos-of-the-conpaas-workshop-are-available/.

**Workshop on Dependability & Interoperability in Heterogeneous Clouds** Euro-Par is one of the major scientific parallel computing conferences in Europe. On August 27, 2013, Christine Morin (INRIA) and Roberto Cascella (INRIA) organized the Workshop on Dependability and Interoperability in Heterogeneous Clouds (DIHC 2013) `http://dihc13.conferences.iit.cnr.it/` in Aachen, Germany, in conjunction with Euro-Par 2013. This workshop, co-chaired by Thilo Kielmann (VUA) and Paolo Mori (CNR), aimed at bringing together researchers from academia and industry interested in the design, implementation, and evaluation of services and mechanisms for dependable cloud computing in a multi-cloud environment. The workshop was attended by about 20 people, who interacted via many questions with the speakers. The papers and keynote presentations generated interesting discussions.

**Contrail Cloud Summer Schools.** The Cloud Summer School 2013 was held in Almere, The Netherlands, on July 22–26, 2013. It addressed high level technology topics in the field of Cloud computing systems and application. In-depth knowledge was offered on topics like Cloud applications, high-performance computing, Platform-as-a-Service, Cloud federations, Infrastructure-as-a-Service. The Summer School featured also several hands-on sessions in technical collaboration with people from EGI.

Each presentation of the Cloud Summerschool 2013 has been videotaped to use it for later training activities. The presentation slides have also been made available at the Contrail project website: `http://contrail-project.eu/programme3`.

The 2013 Cloud Computing Summer school is the successor of the successful Contrail Cloud Computing Summer school 2012 held in Almere, The Netherlands, on July 23–27 and of the Contrail Summer School 2011 held on June 27–July 1 at Giens Belambra club, on the French Riviera. These Summer Schools featured a Doctoral symposium where all registered participants were encouraged to seize this opportunity to exchange with other school participants on their research topics/areas.

**Contrail booths at major international conferences.** The Contrail project presented the project results to a number of booths at international conferences. It is worth mentioning the presence at SC'11 (Seattle, USA) and SC'12 (Salt Lake City, USA) and at ISC'12 and ISC'13.

**EGI Community and Technical Fora.** EGI (European Grid Infrastructure) is an early adaptor community of Contrail. We collaborated intensively together, both on a business development and on a technical level in particular concerning security. Hence we were present at the two major EGI events during the past year. In 2013 EGI organised a community forum in April in Manchester, UK and a technical forum in September in Madrid, Spain. At both occasions, Contrail partners participated to these conferences.

At EGI CF, April 8-12, 2013 in Manchester (UK), Contrail partners presented the Contrail project in a booth, a tutorial, and a presentation on "Federation Security in Contrail" in the Federated Identity Management Workshop. A Contrail tutorial was held on the topic of Contrail Cloud middleware support for EGI community platforms. At the EGI Technical Forum in Madrid (Spain) September 16-20, 2013. Contrail was represented by two presentations on security and interoperability in cloud federations, with the latter at the co-hosted Cloud Interoperability Week.

**OpenStack Summit.** OpenStack is one of the most important Open Source Cloud computing software stack. Contrail participated to the OpenStack Summit, held April 15- 18, 2013 in Portland, Oregon, USA,

with a presentation on "Federation and its Security aspects". The focus of the talk was the open source Contrail software stack and a live demo of the deployment of an application through a Contrail's web portal.

**White Paper.** The Contrail White Paper gives an overview of the Contrail technology developments, especially targeted for business people. It provides a summary of the technology and an overview of the uses cases, in the context of current Cloud computing developments. Two versions of the White Paper have been produced in 2012 and 2014. The second version is a completely updated and expanded version of the first version. It integrates the latest developments and new releases of the different software components, but it also expands on the details of the different use cases, and incorporates the use of Contrail components in other projects.

At the end of the Contrail project the White Paper can be defined as the portfolio of the Contrail system, its components and its usage. Of course, if one wants to dive into the technical details of the Contrail technology, one needs to turn to the training videos, technical Wiki's, development documentation, and scientific papers.

## 4.2 Exploitation of the results

The Contrail consortium has produced the open source Contrail software stack, reaching the maturity with version 2.0 released under BSD (3-clause) license in March 2014, and a number of components that can be exploited and reused independently. In this section we first summarise the main components and then we detail the exploitation impact. Finally we conclude with the contribution to standardisation and open source communities.

**Federation** The Contrail Federation is an open source software federating multiple cloud providers, released under the BSD (3-clause) license. More details about the scientific results achieved with this software are in Section 3.1. After the Contrail project the software could be further maintained and developed in other collaborative projects. CNR is also involved in defining the Italian Cloud strategies for e-government and in the optimization of urban infrastructures via ITC and Clouds along the Smart cities paradigm. Potential impact of Contrail Federation will be for companies offering services and solutions to their customers, for public institutions willing to exploit services over federated Cloud resources, as well as for companies providing data center hardware and outsourced management services to public institution's data centers.

**SLA Manager.** The Contrail SLA management is an open source software enabling automatic multi-round negotiation, with QoS and QoP term support, released under the Apache 2.0 license. The software integrates and extends the results of the SLA@SOI European project and adds new functionalities and capabilities. More details about the scientific results achieved with this software are in Section 3.2. Potential impact will be for customers implementing a private cloud, cloud providers integrating SLA Management in their offer, and consulting support to system integrators. There will be potential reuse of the code in other EU collaborative projects.

**Monitoring.** The Contrail Monitoring is an open source software integrating the monitoring functionalities of OpenStack and OpenNebula. It is released under BSD (3-clause) license and it will be potentially reused in other collaborative projects. The scientific results are detailed in Section 3.2.1. The Contrail monitoring support has proven to be efficient and planned to be transferred into OpenStack, first as extensions of existing BluePrints as well as a plan for adding new BluePrints for the next release of OpenStack.

**ConSec.** The Contrail Security software consists of two main parts that can be exploited independently: the authentication module integrating federated identity services and the UCON authorization module. It is an open source software, released under BSD (3 clause) integrating security functionalities with a specific focus for the adoption of standards. More details about scientific results are in Section 3.3. The software is currently reused in other initiatives, like EUDAT, and there is potential interest by other communities thanks to the support for external Identity Providers. Being split into components, which talk standard protocols to each other, makes it modular - a framework for federated identity management - which helps make it more sustainable and reusable than if it were a monolithic service. There are potential contributions to standards, in particular the Python implementation of Oauth. The UCON authorization module, being a compatible extension of a standard open-source XACML engine, can also be exploited as a drop-in replacement to extend existing information systems with UCON authorization features.

**Virtual Execution Platform.** The Virtual Execution Platform (VEP) is an open source software for IaaS management and distributed application deployment under SLA constraints, released under the BSD (3-clause) license. The software integrates the open source OVF-toolkit and an external scheduler module developed within the Contrail project. The software currently supports OpenStack and OpenNebula IaaS resource systems and further development to add support for the OCCI standard is planned. More details about scientific results of VEP are in Section 3.4.1. The VEP-S EIT ICT Labs activity for 2014 focuses on VEP to improve the software for its technology transfer. The software will be potentially reused in other collaborative projects and in EGI community. Potential impact will be on the market to manage private clouds and/or offer customized solutions on top of VEP.

**XtreemFS.** XtreemFS is an open source software providing a reliable distributed file system, released licensed under the non-restrictive BSD license. The details about scientific results of XtreemFS are in Section 3.4.2. The software will be potentially reused in other collaborative projects or communities, like EGI and the Dutch Health Hub. The impact of XtreemFS in the open source community is high: the latest version of the XtreemFS windows client was downloaded more than 3000 times in 2013. A start-up is commercialising solutions based on XtreemFS.

**ConPaaS.** ConPaaS (Contrail PaaS) is an open source software implementing a platform offering elastic PaaS services over multi clouds. The details about scientific results of ConPaaS are in Section 3.5. The code is released under BSD (3-clause) license and it is currently re-used in several collaborative projects: EU FP7 Harness project and the national Dutch initiative COMMIT. A 2014 EIT ICT Labs activity focuses on ConPaaS to improve the software for its technology transfer. Potential impact will be for small businesses in need of scalable Web applications and scientific communities in need of managing large-scale data.

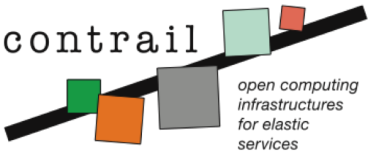
**Scalarix.** The distributed file system Scalarix offers a transactional key-value store service. The details about scientific results of ConPaaS are in Section 3.5.1. It is released under the non-restrictive Apache License 2.0 and has a large open source community of users; the latest version of Scalarix was downloaded more than 750 times since October 2013. The component is also reused in other collaborative projects: Harness and IES-Cities.

## 4.3 Impact in standard organizations

The standardisation activities of Contrail partners have been focusing on participation in different research groups and working groups of the following standard organizations: Open Grid Forum (OGF); Distributed Management Task Force (DMTF); Organization for the Advancement of Structured Information Standards (OASIS); and The Internet Engineering Task Force (IETF).

Contrail uses and implements several standards, and among those the most important are OVF (Open Virtualization Format) from DMTF for application description, CIMI standard proposition from DMTF, OAUTH2 from IETF, and XACML from OASIS. Contrail has also contributed to standards with a Python implementation of Oauth and discussions within the federated security community group of OGF. The details about the standardisation activity can be found in Contrail deliverable D16.5 [32].

# 5 Contacts





| Organisation name | Country |
|---|---|
| Institut National de Recherche en Informatique et Automatique | FR |
| XLAB | SI |
| Consiglio Nazionale delle Ricerche | IT |
| Science and Technology Facilities Council | UK |
| GENIAS | NL |
| Tiscali | IT |
| Konrad-Zuse-Zentrum für Informationstechnik Berlin | DE |
| Hewlett-Packard Italiana s.r.l. | IT |
| Linagora Grand Sud Ouest SA | FR |
| Vrije Universiteit Amsterdam | NL |

### Coordinator

Christine Morin, INRIA
Campus universitaire de Beaulieu
35042 Rennes cedex, France

**Contact:** contrail-contact@inria.fr

**Public web site:** `http://www.contrail-project.eu`