Wireless Pers Commun (2011) 58:31–48
DOI 10.1007/s11277-011-0287-6

# User Centric Community Clouds

**João Paulo Barraca · Alfredo Matos · Rui L. Aguiar**

Published online: 9 April 2011


**Abstract** With the evolution in cloud technologies, users are becoming acquainted with seamless service provision. Nevertheless, clouds are not a user centric technology, and users become completely dependent on service providers. We propose a novel concept for clouds, where users self-organize to create their clouds. We present such an architecture for user-centric clouds, which relies on self-managed clouds based on doctrine and on identity management concepts.

**Keywords** Cloud infrastructure · Identity management · User-centric systems

## 1 Introduction

Technologies associated with Cloud Computing have become quite popular recently. As they bring about the future of distributed computation and services, they come with a mixed pool of advantages and drawbacks that must not be neglected. If we are to let such technologies permeate all aspects of our daily life, our reliance on cloud service providers becomes absolute, confronting us with the realities of these new services.

Clouds indeed provide an invaluable service to the end user and are becoming the next personal servers and computing devices. Users create their environments on cloud services, store their files and most important backups, run publishing services to blogs and websites, and basically run every desired service, since by now, every aspect of home computing has a Cloud counterpart, many running as cheap services. While it requires little technology knowledge to users, these new cloud services introduce unprecedented service conditions,

J. P. Barraca · A. Matos · R. L. Aguiar (✉)
Instituto de Telecomunicações, University of Aveiro, 3810-193 Aveiro, Portugal
e-mail: ruilaa@ua.pt

J. P. Barraca
e-mail: jpbarraca@av.it.pt

A. Matos
e-mail: alfredo.matos@av.it.pt

where availability is usually in the 99 percentile, at low cost, with distributed access, and synchronized across a multitude of devices, including low power mobile devices. In fact, this is simply too much of a good deal for the user to pass up.

Strangely enough, in this environment, the increasingly complex mobile devices, laptops and desktop computers we now have are mere caches of the information present in the cloud. The proliferation of Netbooks, Tablet PCs, and cheaply available smart phones could be seen as a consequence of this, but even those are now powerful computing devices of their own right.

This paper brings up a new view to clouds. We reposition clouds as a technology centered in the user. This view covers multiple aspects of cloud computing, which can be simultaneously or individually considered: the ability of users to explore their devices in self-organized clouds, the ability of users to control its information inside a cloud, and the automated support for increased heterogeneity inside a cloud. All this supported by an infrastructure created in a self-organized way, eventually with resources contributed by users themselves. The next section briefly discusses clouds computing. Section 3 provides a brief snapshot of identity-management aspects, while Sect. 4 discusses how users ARE positioned in current clouds. Section 5 presents our concept of user-centric clouds, while Sect. 6 presents an high-level functional architecture that addresses the major challenges of such a concept. Section 7 then discusses the key aspects of our proposal, while brief conclusions are presented in Sect. 8.

## 2 Cloud Computing

Clouds appeared at a time were most systems were single core and technology was evolving at an ever increasing pace. At this time, most computation was done at large datacenters with monolithic services and where scalability was vertical, that is: scalability required buying more powerful hardware [1]. As a consequence, it was imperative to always have the best hardware installed so that performance of the services provided was maximized. There was also little knowledge on how to build massive distributed computation infrastructures (which were not clusters, not larger mainframes datacenters, and not High Performance Computing environments). Operating systems could not cope with such large-scale distributed operation, and novel middleware, running on top of commodity hardware and software, had to be developed. In the same Cloud there could be several of these distributed entities providing different services such as storage, distributed processing, or document cataloguing or retrieval. Because systems and applications are so diverse, this middleware has to be able to operate on a large set of heterogeneous systems, providing a common set of services to many heterogeneous applications. In many aspects, this middleware replaces operating systems functionalities and provides an abstraction to underlying OS and physical resources. However, in the case of clouds, scalability is horizontal as the support applications can scale by adding more computation hosts (in contrast to vertical which requires upgrading hosts while keeping the number).

Clouds emerged as the new trend for hosting and service delivery for the Internet. NIST defines [2] that Clouds are a model for enabling convenient, on-demand access to a shared pool of configurable resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. Amazon was a pioneer in Clouds when it introduced the Amazon Elastic Computing Cloud (Amazon EC2) service [3]. It provides interfaces (an API) to the existing infrastructure and services, allowing users to use spare disk space, and deploy applications that consume computational resources, thus decreasing off peak capacity waste, and leveraging CAPEX. Users access services through well-known

interfaces, which are supported by a pool of hosts, at an unknown location, and organized with unknown structure (from the user point of view). In fact, current Clouds are massive structures, providing an environment where applications can exist, share data and store information permanently, supported by a myriad of different hardware, ruled by application engines and support services, in a completely transparent manner. Mechanisms such as BigTable, MapReduce, CouchDB, and many others were also created in order to better exploit parallelization, thus leading to better performance.

In this environment, each machine runs its own operating system and some part of the middleware control modules. With the advent of multicore operating systems [4], we may observe a shift to these new operating systems (OS) managing a Cloud, thus putting some functions of the middleware into the OS, with increased efficiency as a result.

Existing cloud deployments are mainly seen as providers of Software, Platform or Infrastructure. The same cloud can be marketed simultaneously in many forms, each generally around one of these forms. Software-as-a-Service (SaaS) represents one of the most common applications for clouds. Through a single interface, and by exploiting multi-tenancy, it is possible to provide a service, which resides in a distributed environment, benefiting from the enhanced data replication, security and scalability of the cloud. Platform-as-a-Service (PaaS) can be seen as a multi SaaS scenario where a complete environment is provided, which can be composed by multiple SaaS instances. PaaS business model focus in renting the entire platform service stack and not individual services, but in essence it is similar. When the cloud provides Infrastructure (Infrastructure-as-a-Service or IaaS), the business model is focused in providing an infrastructure composed by set of nodes (usually a virtualized partition), which can be used to deploy custom services and applications. In this case, the service provided is more focused in availability, redundancy and scalability, rather than application stack.

## 3 User Centric Management

The evolution of the Cloud paradigm has lead to an explosion of services available over the Internet. This followed closely the trend of recent years, where Web 2.0 opened the door to the social web, making users signup and register with different services, blogs, forums, and a wide range of utility applications. As the number of accounts and corresponding passwords grew, a problem emerged facing the authentication and access control of end-users, which now faced multiple accounts and login forms, locked in information silos, resulting in a large amount of replicated information. In this ecosystem, Identity Management concepts emerged as means to relieve the burden of user management, both on the user side (by handling multiple providers and service through a single account, requiring a single sign-on action), and on the service side (by reusing authentication and authorization systems, and enabling federative properties).

The Liberty alliance (ID-FF) [14] proposed a single-sign-on (SSO) platform with federative capabilities, using SAML [15] assertions, while Shibboleth [22] stemming from educational environments, focused on a user-centric approach using identity providers. Both these initiatives lead to the creation of the SAML 2.0 [16] standard, which covers a wide range of problems, such as single-sign-on, federation and attribute exchange, all with a strong emphasis on end-user privacy. The adoption of SAML 2.0 is also reflected on Cardspace [19], a Microsoft effort to create a recognizable paradigm or user identity. On the Internet, OpenID [18], a simpler standard that provides single-sign-on across multiple websites, has become popular. OpenID has seen adoption by providers like Google, converting all their user accounts into OpenID identities, usable at any enabled website. Another open protocol is

OAuth [20], with major social websites like Twitter and Facebook providing OAuth support, enabling their users to login on other services using their Facebook or Twitter credentials. OAuth is now being standardized within the IETF [21].

Clearly, Identity Management (IdM) is becoming an integral part of web services. And the paradigm shift towards identity management is not happening solely on the Internet. The user is becoming increasingly defined as an identity, and identity driven architectures and services are becoming a reality. As part of a research endeavour, the IST SWIFT project [23] has proposed a vertical application of identity, bringing it to all levels of the network stack and not only the service layer. This lead to a user centric network, where identity is the driving force behind network provided services, preserving privacy while providing an unprecedented level of pervasiveness and personalization.

When considering multiple users, especially if they share some attributes, the individual identity can belong to groups or communities. Communities are groups of close-knit persons, united by some common denominator. These differ from standard groups because the common denominator is related to an interest (e.g. music, search for extraterrestrial life, free software), and not to a characteristic of the members (e.g. age, sex, height). Communities, from a social perspective, usually [32] consider issues as membership, influence, integration and fulfilment of needs, and shared emotional connection. Also, basic properties such as tolerance [33], reciprocity [34] and trust [35] are frequently raised as important to communal behaviour. Due to the complexity of managing communities, automated management systems are required, and thus formal representations need to be developed. All the objects (services, users, peers, roles, delegations…) instantiated for management purposes need to be represented in common languages (such as XML), having a structure that has to be in accordance to a well-known ontology, to support the potential complexity of existing relationships. Objects are particular to a given environment and are permanently available to every authorized participant, according to its permissions. The collection of all management knowledge is named the Doctrine (the term Doctrine is usually used because users must believe in its rules, and it's not blindly imposed).

There is a large amount of solutions in the area of autonomic management of interacting actors, mostly focused in the creation of languages and frameworks for autonomic, user centric coordination. Research efforts all point in the direction of a knowledge base where players can access. This knowledge base includes rules and roles where behavior is conditioned by a reasoning engine processing existing objects. Solutions are usually constrained to particular applications but can still be applied to the scenario of user centric clouds - as the foundations for community organization are already deployed. SOUPA [5], targets pervasive applications and identifies the need of ontologisms in order to cope with the heterogeneous nature of pervasive environments. For considering user centric clouds, the heterogeneous nature of the services provided requires methodologies to manage data in flexible manners, which is also further discussed by Zhu et al. [6]. Applied to the networking environment, CoRaL [7], proposes a language and system to manage radio spectrum. While very useful, as it is bound to ontology based reasoning for spectrum management, it is limited to a very specific application, unless the ontology is expanded. Still it presents interesting concepts that can be further applied to other scenarios. KAoS [8] seems to be the one of the most flexible solutions proposed, with applications over many fields. It is highly complex and requires well-defined Distributed Directory Servers (DDS), but this complexity is reflected in the expressiveness of the rules it creates. One problem is that reasoning incurs in a high number of complex queries over many objects of the DDS, thus raising scalability issues. Specific for grid and clouds environments there are many other solutions, which can serve as a basis for the creation of community clouds. In this area we can find the Community Authorization

Service (CAS) (based in the Globus Toolkit Middleware) [9], aiming at defining policies for resources in (virtual) domains, which must be agreed by all domain administrators (we consider that a single user with a single resource is a special case of a domain with only one resource). CAS has a tight integration with IdM and supports standard representation methods such as SAML. With similar functionalities we can also find TrustCom [10], and then GridShib [11]. In the case of GridShib, the aim is to create scientific communities by providing an attribute based IdM based on Shibboleth.

## 4 Users in Current Clouds

The Cloud high availability conditions at near zero cost, along with distributed access and synchronized access across mobile devices, provide tangible value for end-users. The added value of these services that provide a level of service availability, which is simply out of reach for the average person (or even a small corporation), associated with a (mostly) free-to-use business model serves a strong lure for the user. This encourages a large participation, increasing the service value and reach, and creating unofficial "standards" that are assured by its massive use and the entire ecosystem that arises around it.

However, there are hidden costs in this landscape. The price of having ubiquitous service access through heterogeneous mobile devices, and mobile applications, that exchange data with 99.9% service availability at zero cost—good free/cheap services with no hassle-takes its toll on privacy and security. The most important downside is that users have little or no control over their data. It is almost impossible to determine where the data is, who accesses it, how safe it is, and how can it be deleted, or even if it actually gets deleted. In fact, many applications operated by banks and governments require assurances related to the location (and jurisdiction), existence and deletion of particular data, which pushed some players away from clouds [8]. These features were granted in our normal everyday usage of personal computers, a view now misty, since we do not have access to the servers running the Cloud, and these might not even be all on the same location or country. As such distributed systems try to reach millions of users, a single security breach can jeopardize many users, rather than the single or just a few users when a breach occurs on a home or SOHO computer. The same principles that make Clouds attractive (make a service available to all devices, anywhere in the world, for a massive number of users) also raise several new problems. In fact, the business models undermines privacy to a great extent. Free applications and services, in order to survive, must produce profit by selling private user information, even if anonymized. It is attractive for service clouds to profit from user personal data, while providing a free service to users. There is currently no way to establish what exactly happens to that data, and what does it mean when the service says that our data is "protected". And the more the service profits from user data, the more Infrastructure and Platform profit from direct contracts with the service layer. The entire cloud ecosystem is centered on services and providers, neglecting several key user related aspects. Also, after one of these companies goes off business, user data is considered an asset, which can be sold to the highest bidder [12], or transferred at any time when there are mergers, business expansions or simply there is the need for it [13].

The problem of user data is further stressed by the fact that the cloud can span over multiple servers, datacenters and even countries. Who is legally bound to answer before the data stored? If a court should mandate, who applies which law in which jurisdiction? These were problems of our current Internet, that that now got greatly amplified by the current cloud model, which not only covers all aspects of user data and service interaction, but also pushes a single service over several countries, with competing legislation.

Another aspect is that clouds are a distributed model through which a single entity can provide an aggregate services. Users are only viewed as consumers of cloud services. Their role as providers only applies to content (e.g. links in social networks, blog articles or videos) and not to the service provisioning platform or even the provisioned services. If an user wants to provide a service, lets say through a web application, it must rely on existing cloud infrastructures as there no way for him to create its own cloud, or gather efforts in that direction by acting as a magnet for other users. Clouds are massive infrastructures, providing many services, but under control of a small minority of players. As these players become more dominant in the market, and invest more resources in their infrastructures, there is little interest for them to change the game.

## 5 User Centric Clouds

What we propose is a paradigm shift to user-centric community clouds, making the cloud also about the user, and not only about the infrastructure and services. By leveraging Self-Organization and Identity Management, we envision a path where the user takes the cloud concepts and turns them to his advantage. The traditional separations on cloud services as infrastructure-as-a-service (IaaS), platform-as-a-service (PaaS) and Software-as-a-service (SaaS) then can be structured according to slightly different lines (Fig. 1): user owned infrastructure is provided as a service, running a self-organization software able to provide a distributed platform, and supporting user-controlled applications, allowing users to control its information with the requested level of security and privacy.

A cloud system that uses user-centric technology can put the user back in the driver seat of his data, providing the required Data Control, Information Security and Privacy that must come with all modern systems. This can be achieved by designing a system that explores the heterogeneous user-owned infrastructure, in a fully distributed way, but providing distributed privacy, security and authentication; the system then provides OS-alike semantics to access all the existing services and API's, harmonizing application field. However, the current cloud model does not consistently provide these features, and while they could be implemented separately—enhancing current cloud technologies—they would not benefit of user-centric characteristics beyond the fragmented model that exists today.

In order to do this, clouds need to incorporate a different set of technologies. Fortunately, these technologies have been explored in different fields, providing a simpler integration path towards clouds.

- Self-organized user-centric infrastructures, applied to the provision of a distributed heterogeneous infrastructure.

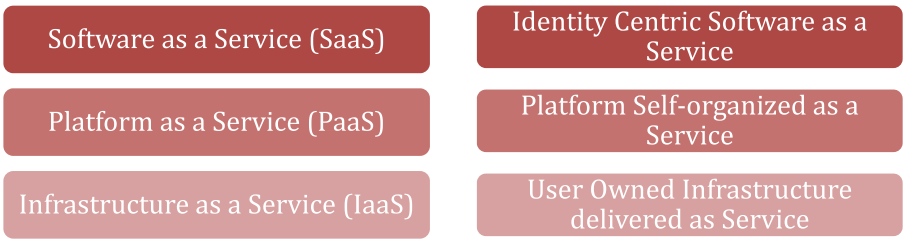


**Fig. 1** Clouds structuring: operator-driven (*left*) and user-centric (*right*)

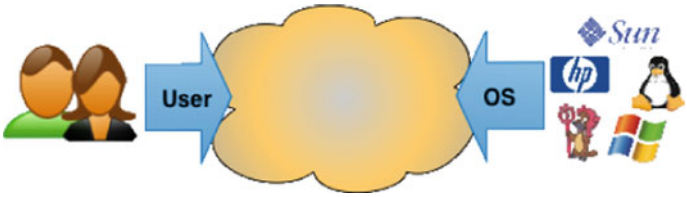


**Fig. 2** A User-centric cloud enables users to cluster around communities of interest providing cloud services

- Cloud operating system semantics: Providing a common API and reusable semantics, while bolstering a tight connection to identity and data management.
- Identity Management (IdM) as a user-centric/driven technology, applied to general-purpose computation and as the main leverage of user data.

### 5.1 User-Centric Self-organized Systems

Many research and commercial efforts exist towards enabling peer-to-peer communities sharing Internet access through their wireless access points for mobile use [24–26]. The vision of Negroponte of a "Wi-Fi 'lily pads and frogs' broadband system built by people for the people" [27], or neighbourhood self-organizing WMNs, according to Microsoft [28], is becoming true in some areas. Although not exploring locality, the same principle was in part used to build large distributed grids used by projects such as SETI and later BOINC (a framework for multiple grids). These applications strengthen the social capita[1] between people living in the same neighbourhood, or sharing similar interests, (and close the gap between virtual and physical communities) by supporting a large variety of—possibly novel—social and collaborative applications. In fact, social networks are the most accomplished example of these collaborative environments, although usually mediated by a social network provider, and all data is centralised in privately owned clouds. Nevertheless, technologies exist that allow sharing of infrastructure in a controlled way by creating multiple mediators [29] or even that allow the creation of distributed services (such as social networks) without a mediator [30]. These systems arise from the need of creating a non-controlled environment where users can freely express their ideas. More importantly, where they can take control over the data presented to others and become active participants in the infrastructure. As with wireless ad-hoc networks, such solutions have the potential of great scalability and the property of adaptation to increasing number of participants (as new participants can potentially add more resources).

In our vision, users can thus consider exposing not only their communication links, not only their files, but their computational infrastructure as well, as a shared infrastructure for the community (see Fig. 2).

### 5.2 Identity as a Cloud Kernel Technology

In a distributed environment running under the same operating system, linking activities to specific users is essential. Processes become distributed, in a distributed infrastructure, with

[1] Social capital is related to the importance of social relations, interaction between diverse people, and the resulting norms, values, ties and relationships. For a list of the currently most accepted definitions and authors please access: http://www.socialcapitalresearch.com/definition.html.


**Fig. 3** Main functionalities provided by an identity management system

multiple ownerships. Identity management technologies play a key role here. While IdM is usually regarded as an application layer technology or even something that can work at the network level, we propose to make it an integral component of the Cloud operating system. This enables designing a user-centric operating system that is build around user data and resources, while at the same time manages a pool of distributed environment resources (the cloud) (see Fig. 3).

Every service running in the cloud operating system will have the notion of user. In fact, it is the user information, through access credentials, that enables service access. By aggregating this information in the IdM system, it is possible to have a broad view over the Cloud resources available to each user, and to provide this view to self-organizing software. From this point on, the IdM layer becomes the single place capable of correctly discovering and aggregating user resources scattered through the cloud in a coherent architecture. This is why IdM needs to become a kernel component of the cloud OS. It is the entity capable of leveraging cloud services, much like today's operating system scan the system bus for plug-and-play devices and make them available to the remaining devices.

Every piece of data and of infrastructure is owned by the user, and identified by its identity. This makes the IdM system responsible for how data is distributed, processed and stored (see). This is already part of modern IdM systems, where Attribute Providers handle secure, private storage of information, becoming a matter of defining appropriate semantics for services as in operating system.

## 6 Architecture

Our concept can be summarized in one simple paragraph. We wrap the traditional Cloud structuring (SaaS/Paas/IaaS) as modified logical layers that are managed by a top level control layer built with IdM technologies (which considers individuals and groups), supported by a lower layer, consisting of a Cloud Operating System (which can be a middleware or a real operating system), that provides self-organized community features. Each one of these blocks can be considered as a separate sub-system, and its implementation(s) may lead to user-centric clouds with different characteristics.

The IdM layer provides user centric mechanisms that are traditionally built on top of existing communication infrastructures (see Fig. 4). Users can self-organize into communities, provide resources, and use Cloud services, taking in consideration trust relations in a

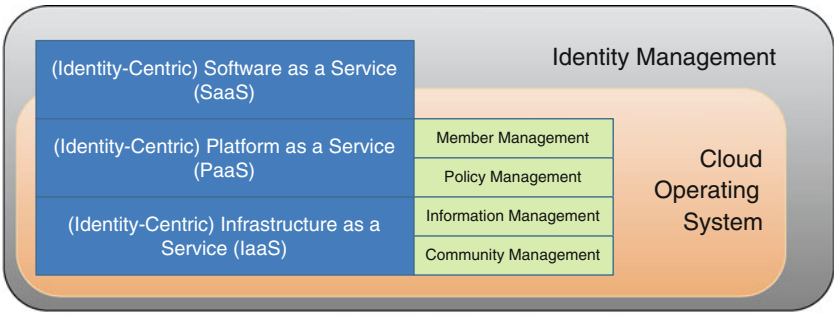


**Fig. 4** Proposed user-centric sub-system blocks

peer-to-peer manner. This enables the creation of distributed rewarding and charging models as well as dedicated service clouds in accordance to user interests. These services are reused, throughout the lower layers of the new model, becoming an integral part of the user-centric Cloud OS. The Cloud OS provides the core primitives and interfaces that enable the tight integration between infrastructure, platform, services and IdM.

### 6.1 Cloud Operating System

In the light of a loosely coupled, distributed architecture, such as the scenario of multiple computation islands providing computational resources (e.g. at user premises), the Cloud OS evolve a multi-core OS (which are gaining high momentum), to a multi-host OS. A key research challenge for such an OS is that host churning rate and latency are higher when considering commonly available network access interconnects. However, trends point to ever-increasing bandwidth and decreasing latency, which will surely reduce execution penalty of running a distributed OS over network access technologies. For current operator-centric Clouds, which operate in tightly coupled deployments, it is expected that OS deployment will be much facilitated.

In our approach, the Cloud is structured along three layers, slightly different from the traditional ones (see):

- **An infrastructure layer**, which will be composed by a multitude of devices and systems, which provide available storage and computation resources to Clouds. Administrative ownership of these devices may be varied and we expect users to take a step and be active participants as providers. We expect many of these devices, even if owned by the same entity, to incorporate multicore systems or even small clusters, inherently heterogeneous. The advent of multicore operating systems will be a major breakthrough for bringing cloud concepts to operating systems, since these will reach a complexity akin to large distributed systems. In the mean time, a middleware can provide resource aggregation capabilities.
- **A service platform**, which be inherently self-organized. Either in a user driven community, or in a cloud cluster, we expect the operating system to restructure itself according to a set of policies, either shared and agreed automatically by users, or imposed by an administrative owner. These policies are translated to task scheduling priority or resource access permission, and are expected to be dynamic, changing over time. It should be noticed that user driven clouds will require more explicitly stated cooperation rules in order to keep performance, security and privacy levels, as well as correctly rewards users

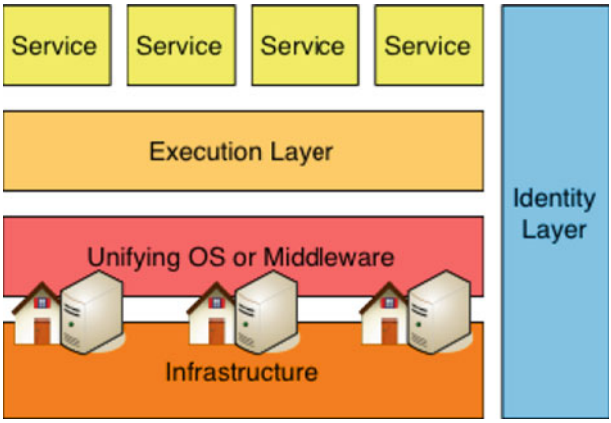


**Fig. 5** An unifying middleware connects different users, while providing a common execution layer for services. User Identity must be present across all layers

for their participation. Each service platform may aggregate many users and provide one or more execution environments for clouds applications.

- **A software execution layer**, where distributed applications will run, under constrain of a strong IdM system, and with mechanisms limiting access to computational and information resources according with the access control rules and system policies. More importantly, execution should take in consideration the owner of each instance, its location and past behaviour in order to better optimize system usage (see Fig. 5).

### 6.2 IdM as the Glue Layer

The intended outcome of using IdM as the driving technology is to give users the control of services and cloud information. By providing IdM as the glue layer between services and their interaction, we enable intrinsic user-centric mechanisms, especially concerning authentication and access management, along with privacy control of sensitive user information. Moreover, IdM allows the creation of long-lived trust relations between digital entities, and the creation of a reputation system allowing for distinct access control service provision taking in consideration provided resources and past behaviour. We can describe the IdM sub-system as a set of services, which must have well known interfaces, in order to be reused throughout the entire architecture:

- **Authentication:** Provide user authentication that enables building user centric services and mechanisms, throughout the entire architecture. This enables services and Cloud OS to build secure and controlled environments that protect user privacy.
- **Access Control:** Resource and service management through proper permissions, created and managed around the notion of a representation of the user identity valid in the system. Access control also considers user reputation, past behavior, and participation rules when providing access to a resource.
- **User Information:** Attribute and data exchange that enable services to obtain and use user information for federation purposes along with personalization of cloud services.
- **Privacy Management:** Policy based management of user attributes and private information, tightly integrated into the data exchanges within the architecture. This module

considers personal user data (such as identifier, contacts), community data (trust relations, membership, reputation), as well as service and user specific data.

- **Security Primitives**: Key management for cryptographic operations, across different services. Data sharing rules must be expressed in the cloud and available to all participants, and should be enforced when exchanging data with external entities. Because system operates over a multi-owned infrastructure, strong cryptography must protect all data.

When combined with strict interface definitions, the above-mentioned logical functions compose a kernel block of the proposed Cloud OS. These requirements convert any and all user-related functions into standardized identified operations that are carried out within the cloud. From the reusable functions stems the cross-layer properties that characterized the proposed IdM approach. This promotes that, like in common operating systems, every application or operating system client goes through the same common functions and libraries to access the operating systems semantics, which in our case are entirely IdM based.

Note that most efficient IdM solutions (such as the ones identified previously) can be adopted. Current cloud environments, and new user-centric designs, should consider the existence of an always-available version of the user identity, as well as the behaviour, access and execution rules in the target cloud environment. We stress that clouds should embrace these mechanisms and provide near-POSIX interfaces in the line of traditional operating systems, supported by a rule based knowledge layer.

### 6.3 Community Self-management

User driven cooperation requires the creation of structures, such as communities, which allow users to focus cooperation around a relevant purpose. Must like existing Internet grids where users can choose to donate their resources to fight cancer or search for extra-terrestrial life, we envision the creation of communities providing clouds for different purposes. The important fact here is that this new clouds are driven by user requirements (which can also be monetary), in opposition to business strategies.

Communities mostly resemble autonomic cooperating systems providing aggregated services to external entities. Their functioning has its roots in the fulfillment of needs (both of members and of external clients), supported by the existence of a knowledge base that is available to all members (a view of this knowledge may be provided to clients). This knowledge base dictates the behavior of existing sub groups, Roles and Rules. Access to some objects (part of the knowledge) can be restricted in the same way access to resources can by the existing rules. Communities are different from systems managed in a central system because users have more control over the system and the knowledge evolves as the interactions between users occur (that may affect its reputation). Moreover, the system is dynamic (both in terms of structure and operation), containing modules responsible for optimizing operation towards predefined goals. These goals are part of the knowledge base that is available to users, making a clear statement over the purpose of the community at any given time. This is done in the form of Roles, which users can enroll into, and Rules describing the expected behavior. In this concept, the community is the result of some users agreeing in a goal and participating in some action, under the Rules of behavior (note that this can be easily adapted if "users" are simply an internal structural division in an operator-driven cloud provider). The community control loop inherits its architecture from the autonomic manager proposed in [31], but considers that knowledge can take in consideration user context and past interactions (see Fig. 6). This loop is required due to the loose nature of autonomic distributed systems. Specific configurations cannot be devised for systems with unknown number (or nature) of nodes.

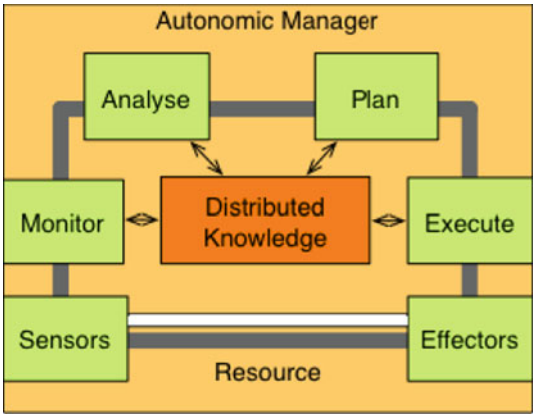


**Fig. 6** Autonomic manager as part of the autonomic control loop

Instead, the research community long agreed in the use of reasoning engines and ontologisms.

The dynamic knowledge repository contains behavioral Roles and Rules. These make possible to have a tight control over the end service provided as well as membership of the community. Rules can state minimum requirements for membership, service access or even service provision. Users with insufficient resources may be denied participation - as its inclusion would decrease the quality of the service provided. Continuous monitoring over the services provided and contributed resources allows for optimum service. Moreover, Roles can be constructed in such as way that users may have to select one particular Role, thus resulting in a very specific service profile (e.g. may provide storage but not VoIP handling). By creating Roles, excluding conflicting services from operating at the same host (e.g. heavy bandwidth consumed by storage services affects VoIP), a proper service can be achieved without the need for carefully planned deployment. We consider that the community should create and adapt Roles according to the goals it aims to, however if the reasoning capabilities of the management software are enough, it may be even able to generate these Rules and Roles automatically, after it identifies that the quality of the service provided can be improved.

The incorporation of community self-organization mechanisms in the Cloud OS should result in the creation of primitives allowing users to create, announce, discover, join, leave or destroy communities, as well as participating in a distributed environment comprised of several hosts at remote locations. In every infrastructure one or more clouds could be provided, each exporting available resources as services either to external clients or members, according to the community policy. As an example promoting user information privacy, each community can define keys for controlling data storage, or system access. These keys, which are secured and only have a local reach, can be used by services to cipher data inside the cloud. Backups and other data maintenance operations are not affected and can still be performed - because data is secure as they operate over bulk ciphered bytes as well as native data (if we exclude higher entropy which usually results from ciphering data). By using a different key for each service, access to others data, and privacy violation, would be highly more challenging, but more importantly, users could permanently invalidate data (including replicas and backups) simply by invalidating the respective key. As information and keys are tied to a specific community, destroying a community would result in effective invalidation of all data.

## 7 Discussion

This user-centric cloud model potentially impacts new business models for several parties, and hopefully, rebalances the role of the user atop the shared infrastructure.

### 7.1 Identity in the Cloud

Regarding Identity as a kernel structure of the Cloud OS implies taking Identity concepts a step further. Besides an advanced authentication mechanism, IdM can become the key driver for distributed cloud services provided by user communities. First, it can become the primary mashup mechanism, by aggregating a user centric view of user and services, enabling the customization and adaptation of services to user needs—a service composition layer on the SaaS level. This is supported by the SSO mechanisms that stem from IdM, allowing a quicker and more reliable composition of Cloud Services into a customized user-centric view, as well as the trusted association of different users into communities. In the process, user attributes can remain private, and IdM can be a cloud building block without compromising privacy. More than the support for communities (with representation of ones' self inside these communities), the IdM could provide further support to the creation of user centric clouds at all (I/P/SaaS) levels.

Dynamic federation mechanisms enabled by IdM can also play a major role in cloud evolution. AAA mechanisms can be established on the fly, based on user-preferences across multiple communities and services. More importantly, user-preferences should always take in consideration the expected behavior of the user communities, and ultimately, interactions will be conditioned by the roles of the communities the user belongs to. In the process, providers can personalize services according to user user-centric information, such as available permissions and requirements, through policy based mechanisms or even Distributed Role-Based Access Control (DRBAC). These mechanisms could modify the way we see clouds, converting them from static service-oriented constructions, into a fluid user-driven paradigm, inherently enabled by cloud operating system semantics.

With this paradigm shift we can finally transform the only static component of the cloud: the user. A rich dynamic mechanism as part of the core cloud operating system allows scaling the services, computing power, storage requirements, permissions, prices and many others, according to the needs of the end-user. In fact, we are scaling the Cloud around User Identity, turning IdM into the de-facto control plane for clouds, always built around users and how they group.

As the Identity Provider (IdP) is responsible for user authentication, and acts as the guardian of user privacy and information, it becomes the primary legal connection between network (services), the communities and the end-user. This enables defining a user as belonging to the legal circles drawn by the IdP, breaking down the complex barriers that scattering information across multiple cloud providers, communities and countries can generate. Furthermore, since the identity can be cryptographically enabled through keys stored at the IdP, it is possible to define encrypted and secure storage across multiple domains, safeguarding user information, and making it only useful when accessed through the Cloud OS, with proper permissions—establishing an equivalent to widespread encrypted hard-drive technology. In this scenario, all information exchange mechanisms should be properly authorized by the end-user, respecting privacy requirements, at the light of the interacting communities, and removing the secrecy and misuse that now haunts private user records. This enables a new dimension to Cloud security and privacy, and turns the control of private user data, to the rightful owner—the user.

### 7.2 Cross Layer Community Awareness

Communities are the aggregation mechanism through which users cluster. Through IdM and a strict RBAC policy system, it is possible to develop an autonomic system capable of providing aggregated services. Interactions take in consideration not only user identity and service preferences, but also operational aspects specified by the community policies. Different communities will have different purposes and goals, thus resulting in very different set of rules and roles. Users belonging to multiple communities will be faced with the fact that policies from the different communities can be incompatible.

Collision can occur at the resource provision level if users overprovision the resources they give to the community (e.g. a 200 GB harddisk cannot be fully allocated to two separate environments; the same happens for computation cycles or memory). The resource management of the Cloud OS must take in consideration the resources allocated to each cloud, and the resources allocated to each community so that conflicts are avoided. This will limit participation, as users resources are always finite.

Policy collision can also occur at the semantic level, independently of the resources allocated by the user. Taking in consideration the example given before, provision of a service (storage for instance), can collide with the provision of a VoIP encoding or call handling service. Inside the same domain, policies can be put restricting users to provide only one. The same become more difficult over multiple policing domains. By enabling federation, through the IdM system, it becomes possible to condition behavior across multiple policing domains.

Both types of policy collision will have impact in the control loop as the services provided by an user under colliding policies will provide a degraded service. We believe that, although federation between all policing domains will be required (especially because an IdM system lays the foundations for federation), the autonomic control loop will be able to avoid over provisioning or policy collision, if proper monitoring rules are deployed. Moreover, by incorporating user identity in all layers of the cloud and services, one could expect that either business or social incentives (depending in the nature of the community) can be put in place, in order to foster optimal behavior [36].

### 7.3 Usage Examples

In the light of User Centric clouds, the community concept presents the opportunity for users to rapidly create self-managed clouds that can be tuned towards the interest of all participants, while being provided by user owned hardware. We could observe the appearance of clouds for distributed storage by a set of friends or a local neighborhood community. These clouds, formed in an autonomic manner (easily supported by the Cloud OS primitives) can provide services to their members and foster the development of applications. But more importantly, an autonomic management system focused towards the creation of clouds, and taking in consideration IdM can be used for the rapid creation of micro-cloud providers composed by users interested in possible revenue. Deploying a cloud infrastructure is way out of reach for most users, however many computational resources are wasted in idle cycles, and storage mediums (mostly hard disks) are easily larger than really needed by a common user. A cloud is composed by many individual systems, which can perfectly be provided by users, eventually returning some profit to its owners. This is a model much used today in the production of electricity for the public grid, where anyone can product electricity and sell it to the public distributor. The main difference for user centric clouds is that users are not

tied to any operator and because the Internet is a transparent pipe, they can create a Cloud environment over the Internet in a autonomous way.

Another aspect is that we can expect the creation of communities-in-a-cloud, which contain other clouds. As hardware becomes cheaper, the increased flexibility and business opportunity of this additional level of abstraction will become much appreciated. Today we already observe providers which sell a cloud (mostly as a SaaS) but which are actually a rented slot in another (PaaS) Cloud. Because they are able to create value with their services, creating clouds inside clouds is now a reality, and our proposal should be able to reinforce this trend.

Software-as-a-Service (SaaS) represents one of the most common business scenarios for clouds. Through a single interface, and by exploiting multi-tenancy, it is possible to provide a service, which resides in a distributed execution environment. IdM is solely applied to these applications when users login into the system, and in order to select the appropriate working dataset. The credentials provided by users are verified with the identity provider (which can be co-located or remote (e.g. using OpenID)). From this moment on, the identity is used to personalize the interface, and load the actual personal data: access to data, and actual processing is done without taking in consideration user identity, permissions, and priority. That is, data is accessed in the same manner for all users, only the working dataset changes. If the application frontend is exploited, users will be able to access others data, specially since data is seldom ciphered. For effective IdM support, and especially because clouds are such a heterogeneous and complex environment, data stored in databases or other permanent storages should be secured by cryptographic means. Because we consider the creation of autonomic communities, which are capable of providing Cloud services, it is even more important the usage of data ciphering—due to the more loosely attached underlying architecture. Communities can also play an additional role in terms of access permissions. As users may also present information related to the communities they belong, access control and even data privacy rules can take in consideration the source and destination communities. This aspect enables fast, autonomic differentiation based on SLAs between clouds, all supported by an IdM platform.

Process scheduling is also a field where IdM could provide benefits. When users trigger some action, scheduling of the new task will not take in consideration user identity and will be executed, at the very best case, taking in consideration aggregated classes (e.g. core services, premium users, standard users, non-paying users, etc…). The result is that when systems become slow or irresponsive, this fact will affect everyone, while it should start affecting lower priority users first, or at least preserve critical users. IdM can solve this by making identity accessible at all layers of the application, including storage access, database access, or network communications. For this, user identities must have a locally valid identifier, which is tied to an ACL or DRBAC object in the community, which has definitions for all sub-systems. User identity must be available to all services, and then mapped (by the actual enforcement mechanism of the destination system) to prioritization of actions, and access permission (e.g. using XACML [17]). This is similar to the case of a traditional OS, where access to services is always verified against the effective user and group id, and scheduling also takes these values in consideration. As another example, users that are members of the community and provide resources for the clouds it contains should be degraded only after other external users are degraded. Taking in consideration the reputation and membership of each identity (mapped in the IdM) is possible to enforce this differentiation.

PaaS can be seen as a multi SaaS situation where a (more) complete environment is provided, which can be composed by multiple SaaS instances (storage, database, indexing, etc…). In a community-operated cloud view, PaaS correspond to a scenario where members of the community provide multiple services, and the composition provides an useful

platform. In the case of IaaS, and because user provided equipment will frequently not be dedicated to Cloud provisioning, the infrastructure will be based in virtualized hosts.

In PaaS scenarios, IdM is more present as applications tend to be simpler (they are databases, storage systems, etc…) and not composite application such as webmail frontends or social networks (which also have storages and databases but are hidden from suers). IdM when applied to a PaaS environment provides the same benefits as applied to SaaS (link between execution/access to identity), because it should be applied to all individual services. However, PaaS over community clouds requires IdM support at a greater extent because PaaS may imply the creation of a partition in the cloud infrastructure, personalized to the case of a particular user (Platform Virtualization). Access control, execution and even privacy should be tied to the identity, and in this case also resource visibility. Without IdM and the capacity for creating partitioned environments, users can try to subvert the platform by exploiting other existing applications (eventually from other clients).

In all situations, user resource contribution to the provided service, and the performance of the resources provided must also be taken in consideration. By effectively identifying which resources are provided, and what is the usage of those resources, it becomes possible to reward users based on actual service consumption (using some currency or reputation). Monitoring service performance is vital in order to monitor compliance for the community rules and for optimizing overall service operation.

## 8 Conclusions

This paper puts forward a new concept for clouds: user-centric clouds. This concept can be interpreted in such a way that the traditional IaaS, PaaS and SaaS concepts are changed in order to incorporate aspects as user-owned infrastructure, operator-less cloud platform, and identity centric information processing and storage. Our approach allows the connection of user requirements with cloud and network providers. In fact, in this concept, different players may align their interests through the negotiation of matching policies. A common framework may automatically expand clouds, according to the appearance of new infrastructure. In fact, such an automated environment allows the easy scaling of IaaS, PaaS and SaaS, either user-centric, or operator-centric. Furthermore, this scaling covers not only increased performance or decrease operational costs, but is much more complete, allowing for the integration of new usage requirements, and of increased flexibility in the use of cloud resources.

With an underlying identity layer, a self-organized cloud operating system can be built, coping with infrastructure heterogeneity, and deploying a truly distributed multi-core system support. The flexibility of such an approach, and the inevitability of many of its characteristics for future systems, highlights the merits of this user-centric cloud architecture.

## References

1. Farland, A. (2006). eBay Infrastructure—A prototype for the future, the Clipper Group Observer, Technical Report TCG2006097, November 2006.
2. Mell, P., & Grance, T. (2009). The NIST definition of cloud computing v15. National Institute of Standards and Technology, Information Technology Laboratory, July 10, 2009.
3. Amazon Web Services, Amazon Elastic Compute Cloud. http://aws.amazon.com/ec2/, as in October 2010.
4. Wentzlaff, D., Gruenwald, C., III. Beckmann, N., Modzelewski, B., Kevin A., Youseff, L., et al. (2010). An operating system for multicore and clouds: Mechanisms and implementation. In *Proceedings of the 1st ACM symposium on cloud computing (SoCC '10). ACM,* New York, NY, USA.

5. Chen, H., Perich, F., Finin, T., & Joshi, A. (2004). SOUPA: Standard ontology for ubiquitous and pervasive applications. In *The first annual international conference on mobile and ubiquitous systems: networking and services, 2004. MOBIQUITOUS 2004* (pp. 258–267). 22–26 Aug. 2004.
6. Zhu, J., & Wang, W. (2008). New-knowledge-view based ontology cloud model. In *International conference on 2008 international conference on computer science and software engineering*, (pp. 1140–1143).
7. Elenius, D., Denker, G., Stehr, M.-O., Senanayake, R., Talcott, C., & Wilkins, D. (2007). CoRaL–Policy language and reasoning techniques for spectrum policies. In *Eighth IEEE international workshop on policies for distributed systems and networks, 2007. POLICY '07* (pp. 261–265). 13–15 June 2007.
8. Moreau, L., Bradshaw, J., Breedy, M., Bunch, L., Hayes, P., Johnson, M., et al. (2005). Behavioural specification of grid services with the KAoS policy language. In *IEEE international symposium on cluster computing and the grid, 2005. CCGrid 2005* (Vol. 2, pp. 816–823, 9–12) May 2005.
9. Pearlman, L., Welch, V., Foster, I. et al. (2001). A community authorization service for group collaboration. Presented at the *IEEE 3rd international workshop on policies for distributed systems and networks, monteray,* California, USA.
10. Dimitrakos, T., Laria, G., Djordjevic, I. et al. (2005). Towards a grid platform enabling dynamic virtual organizations for business applications, presented at iTrust 2005, Oxford, UK.
11. Welch, V., Barton, T., Keahey, K. et al. (2005) Attributes, anonymity, and access: Shibboleth and globus integration to facilitate grid collaboration, presented at *4th annual PKI R&D Workshop*.
12. Gellman, R. (2009) Privacy in the clouds: Risks to privacy and confidentiality from cloud computing, World Privacy Forum, Technical Report, February 2009.
13. Amazon.com Privacy Notice, online: http://www.amazon.com/gp/help/customer/display.html/102-5642892-3545759?ie=UTF8&nodeId=468496, as in October 2010.
14. Identity Federation Framework (ID-FF), Liberty Alliance ID-FF 1.2 Specifications, Liberty Alliance. http://projectliberty.org/resource_center/specifications/liberty_alliance_id_ff_1_2_specifications/, as in October 2010.
15. W3C. OASIS SAML V2.0 Specification. http://www.w3.org/2001/sw, as in October 2010.
16. Cantor, S., Kemp, J., Philpott, R., & Maler, E. (2005). Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. http://docs.oasis-open.org/security/saml/v2.0/, March 2005.
17. eXtensible Access Control Markup Language (XACML) Version 2.0, OASIS standard, February 2005.
18. OpenID. http://openid.net.
19. Microsoft CardSpace. http://microsoft.com/net/cardspace.aspx.
20. OAuth. http://oauth.net/, as in October 2010.
21. Hammer-Lahav, E. (ed.) (2010). The OAuth 1.0 Protocol, Informational RFC, RFC 5849, IETF, April 2010.
22. Shibboleth platform web site. About Shibboleth. September 12, 2009. http://shibboleth.internet2.edu/.
23. IST SWIFT Project. SWIFT Global Objectives, January 8, 2008. Available at: http://www.ist-swift.org/content/view/13/29/.
24. FON. http://en.fon.com/, as in October 2010/.
25. Efstathiou, E. C., Frangoudis, P. A., & Polyzos, G. C. (2006). Stimulating participation in wireless community networks. In *Proceedings of IEEE INFOCOM 2006*, Spain.
26. Antoniadis, P., Le Grand, B., Satsiou, A., Tassiulas, L., Aguiar, R. L., & Barraca, J. P., et al. (2008). Community Building over Neighborhood Wireless Mesh Networks. *Technology and Society Magazine, IEEE, 27*(1), 48–56.
27. Negroponte, N. (2002). Being wireless. In *Wired Magazine, 10*(10). http://www.wired.com/wired/archive/10.10/wireless.html, as in October 2010.
28. Microsoft Research. Self-Organizing Neighborhood Wireless Mesh Networks. http://research.microsoft.com/mesh/.
29. Anderson, D. P. (2004) BOINC: a system for public-resource computing and storage, 2004. In *Proceedings of the fifth IEEE/ACM international workshop on grid computing* (pp. 4–10), 8 Nov. 2004.
30. DIASPORA: The privacy aware, personally controlled, do-it-all, open source social network, online: http://www.joindiaspora.com/, as in October 2010.
31. Kephart, J. O., & Chess, D. M. (2003). The vision of autonomic computing. *Computer, 36*(1), 41–50. doi:10.1109/MC.2003.1160055.
32. Audeh, M. (2004). Metropolitan-scale Wi-Fi mesh networks. *IEEE Computer, 37*(12), 119–121. ISSN: 0018-9162.
33. Walzer, M. *On toleration*. New Haven: Yale University Press.

34. Putnam, R. D. Bowling alone: The collapse and revival of the american community. Simon & Schuster (Eds.), New York: ISBN: 0-7432-0306-6.
35. Coleman, J. *Foundations of social theory,* Belknap Press, ISBN-978-0674312265.
36. Fehr, E., & Gachter, S. (2000). Cooperation and punishment. *American Economic Review, 90*(4), 980–994.

## Author Biographies

**João Paulo Barraca** received the Licenciatura degree in Computers and Telematics Engineering from the University of Aveiro, Portugal in 2004, and 2 years later a Masters degree in Electronics and Telecommunication Engineering from the same university. He joined Instituto de Telecomunicações in 2003, were he develops his research activities, and in 2008 he became an Invited Lecturer at the University of Aveiro. In 2007 he started pursuing a Ph.D. in Computer Science. His research activities include community-oriented autonomic networks, development of experimentation facilities for wireless environments, and networking protocol stacks for multicore environments. He has coauthored more than 20 publications, published in journals and conference proceedings.

**Alfredo Matos** received his Licenciatura diploma from University of Aveiro, Portugal in 2005, and has just concluded his Ph.D. on privacy issues at the same university. Since 2005, he has been a research assistant with the Institute of Telecommunications at Aveiro, Portugal. In the past, he has worked on mobility at the network layer, focusing on hierarchical and fast mobility, along with identifier and locator ambiguities. His current interests focus on providing privacy at alls layers, layers along with cross-layer identity support in next generation networks and systems.

**Rui L. Aguiar** received a Ph.D. degree in electrical engineering in 2001 from the University of Aveiro, Portugal. He is currently a professor at the University of Aveiro and an adjunct professor at the INI, Carnegie Mellon University. He is leading a research team at the Institute of Telecommunications, Aveiro, on next-generation network architectures and systems. His participation in European cooperative research is extensive. His current research interests are centered on the implementation of advanced new networks and systems with special emphasis on QoS and mobility aspects. He is a member of ACM and a senior member of IEEE. He has more than 250 published papers in those areas. He has served as technical and general chair of several conferences, such as ICNS'05, ICT'06, ISCC.