Future Generation Computer Systems 29 (2013) 370–380

Contents lists available at SciVerse ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

CloudSNAP: A transparent infrastructure for decentralized web deployment using distributed interception

Rubén Mondéjar *, Pedro García-López, Carles Pairot, Lluis Pamies-Juarez

Department of Computer Engineering and Mathematics, Universitat Rovira i Virgili, Spain

ARTICLE INFO

Article history:
Received 1 November 2010
Received in revised form
4 August 2011
Accepted 26 August 2011
Available online 17 September 2011

Keywords:
Cloud
P2P
Interception
Web
Deployment
Middleware

ABSTRACT

Over the last years we have seen the proliferation of many new popular web applications, which are commonly used on a daily basis by most of us. The challenges that have to be overcome by web application designers include how to make these applications support as much concurrent users as possible, without degrading application's performance, and without single points of failure. Such complex task would be much easier to achieve if designers could concentrate on the application functionalities without worrying about its wide-area scope and derived problems.

In this article, we introduce CloudSNAP, a decentralized web deployment platform. CloudSNAP allows transforming any actual web application into a globally-enabled and scalable one. By using a distributed Peer-to-Peer (P2P) Cloud interception middleware, all necessary functionalities are injected into existent web infrastructures in a transparent way. Therefore, CloudSNAP provides many benefits from P2P Cloud computing, like a decentralized deployment environment as well as a set of distributed mechanisms, like load balancing, fault tolerance, dynamic activation, persistence and replication. Moreover, our solution offers important advantages: (i) a high degree of transparency and decoupling in all provided services by means of distributed interception techniques, and (ii) the direct deployment of existent Java Enterprise Edition (Java EE) applications and services with practically no changes on them.

In summary, CloudSNAP makes it easy to deploy any Java EE web application into a P2P Cloud infrastructure, and immediately benefit from all of its inherent services at a minimal development and infrastructure cost.

1. Introduction

Cloud systems are taking the enterprise world by storm due to their inherent advantages. First of all, they promise a reduction in management and administration costs of hardware infrastructures. Many companies are considering to offload these management tasks to Cloud providers.

Another important reason for adopting Cloud systems is their ability to scale rapidly in fast-growing environments with massive workloads. Achieving flexible scalability is a complex problem which involves many issues namely load balancing of resources, replication and consistency policies, and horizontal scalability of persistent data stores.

Furthermore, companies prefer not to invest their resources in handling such complex scaling issues thus demanding the maximum transparency in the migration of their legacy applications to the Cloud. Such transparency can considerably reduce their migration costs and thus overcome the arguments against a costly transition to Cloud platforms.

The traditional solution in Cloud settings is to obtain transparency by means of virtualization techniques. Yet virtualization is a coarse-grained approach that may not easily adapt to specific problems of legacy applications, like persistence, performance bottlenecks or fine-tuned load balancing.

In these cases, Cloud providers must offer platform frameworks with specific APIs for Cloud services like Microsoft Azure [1], Amazon EC2 [2] or Google App Engine [3]. Nevertheless, framework development is costly, and therefore another alternative to virtualization is emerging in Cloud frameworks: software interception. By intercepting or replacing software libraries of legacy applications, Cloud providers can get closer to complete migration transparency. Two examples of these transitioning technologies are Amazon RDS (Relational Database Service) [4] and Microsoft SQL Azure [5].

Another recent trend in reducing costs is to combine or assist the Cloud with desktop peer resources. These hybrid solutions (P2P Clouds) combining P2P and Cloud architectures may drastically reduce costs to many companies while achieving increased scalability. Examples of this recent trend may be [6] or [7] among others. By combining the best of both worlds, this new type of Clouds acquires a set of new complementary properties.

* Corresponding author. Tel.: +34 977 558 510; fax: +34 977 559 710.
E-mail addresses: ruben.mondejar@urv.cat (R. Mondéjar), pedro.garcia@urv.cat (P. García-López), carles.pairot@urv.cat (C. Pairot), lluis.pamies@urv.cat (L. Pamies-Juarez).

0167-739X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2011.08.013

- *Cost effectiveness*: P2P Clouds are much more cost effective because the massive desktop computing, storage and bandwidth resources may be aggregated to the more stable Cloud farms.
- *Easier scalability*: scalability is easier to achieve since more edge nodes can easily be added to the working infrastructure.

In this article, we present CloudSNAP, which lies within the P2P Cloud scope commented earlier. CloudSNAP consists of a large-scale web application deployment framework, for easily transitioning web servers and their applications and services from a client–server model to large-scale environments with minimal changes.

By means of a distributed interception middleware [8], we are able to implement the necessary distributed concerns (e.g., load-balancing) to build a transparent infrastructure for web systems. The main contributions of our proposal are:

- the utilization of interception-based mechanisms to provide scalability to traditional web servers thus making web applications and services available worldwide.
- A transparent infrastructure that enables direct deployment of standard web applications and services, with practically no changes. We leverage legacy Java EE compliant applications and thus smooth the transition to Cloud settings.
- A CloudSNAP open source prototype based on P2P technologies that supplements peer or Cloud resources, thus enabling massive scalability.

The rest of the article is structured as follows. In Section 2, we review the research background in this area. In Section 3, we present an initial and brief description of previously related projects. Following, we explore CloudSNAP design, implementation, and experimentation in Sections 4 and 6. Finally, we draw some conclusions and an outline of future research.

2. Related work

To the best of our knowledge CloudSNAP is the first middleware platform to provide transparent Java EE [9] application deployment on a P2P Cloud infrastructure. Therefore, in this section we are going to analyse related products whose main aim is to provide solutions to web application scalability, load balancing, and fault tolerance.

2.1. Client–server approaches

There exist different kinds of approaches that modify the behaviour of web servers in order to provide new functionalities (e.g. scalability). Some examples in this area include server plugins, filters, or wrappers.

- We can find a wide variety of server plugins (e.g., load-balancers, proxies, access control or logging plugins) that directly depend on the target server implementation. Usually, to use these new functionalities, a more or less complicated procedure is needed in order to bind them to a specific server.
- In relation to the filtering approach, Java EE was introduced for instance Servlet Filters [10]. A filter dynamically intercepts requests, before a Servlet is reached. Responses are additionally captured after the Servlet is left. This interception mechanism may transform the content of either a request or a response.
- Finally, ad-hoc clustering approaches (e.g., WADI [11]) aim to solve problems related to state propagation in clustered web servers. Thus, using such alternative allows the clustered server instances to share their clients state in a unified manner.
  Since all of these scenarios are mainly implemented by using server plugins, their main drawback is that they need wrapping extensions for each different server implementation and forthcoming versions.

As observed, all of these solutions are rather intrusive, and by being client–server oriented, none of them is capable of providing a suitable and transparent enough scenario for large-scale web availability and deployment.

2.2. Peer-to-peer approaches

P2P alternatives in this area include dynamic content management solutions for web hosting (e.g., [12]) or structured P2P content distribution networks (e.g., [13]) that can handle static content.

YouServ offers web hosting and content sharing over a P2P network of personal web servers. Although they focus on static content publishing, they also provide a lightweight plugin architecture for constructing small applications. However, YouServ provides a limited proprietary model for dynamic web applications since they are focused on static content distribution.

On the other hand, Coral is a structured P2P content distribution network, which allows a user to run a web site that offers high performance and meets huge demand. It uses a P2P DNS layer that transparently redirects browsers to participating caching proxies, which in turn cooperate to minimize load on the source web server.

CloudSNAP aim goes beyond these solutions, since its goal is to provide availability in a transparent way to web applications which may include static and dynamic content. Therefore, by means of distributed interception techniques, CloudSNAP offers its services without changing the web application behaviour in any way.

2.3. Cloud computing approaches

According to the widely used Cloud definition by NIST [14]:

> Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Cloud technologies are relatively young since Amazon produced a considerable disruption in the Market in 2006 when it began offering Amazon Web services. In fact, Amazon has become a de facto Cloud standard and many companies such as Rightscale provide solutions in a very active enterprise ecosystem. An alternative standard API to Cloud resources is OCCI [15] (Open Cloud Computing Interface). OCCI also provides RESTful web interfaces to the resources provided by the Cloud. Both OCCI and Amazon Web services are mostly focused on the infrastructure resources (IaaS, Infrastructure as a Service), while our approach is targeting the programming platform (PaaS, Platform as a Service).

CloudSNAP can therefore be considered as a *P2P Cloud PaaS solution for Java EE applications*. Also according to NIST, in PaaS:

> the capability provided to the consumer is to deploy onto the Cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider.

In our case, consumers can deploy web applications and services conforming to the Java EE standard onto our distributed infrastructure (see Fig. 1). This clearly precludes vendor-locking since the Java EE standard is widely used for web applications. This paves the way to the adoption of legacy applications and thus easing the transition to Cloud settings.

Unfortunately, to the best of our knowledge there is no other P2P Cloud PaaS solution like the one provided by CloudSNAP. P2P technologies and concepts are being applied successfully to Cloud Software stacks in order to achieve scalability. In this sense, key-value stores like Scalaris [16] or Riak [17] bring concepts from Distributed Hash Tables (DHTs) and apply them to Cloud scalable resources. However, they are mainly used as scalable data stores and not as full-blown PaaS application deployment platforms.