Towards a Federated Cloud Ecosystem: Enabling Managed Cloud Service Consumption

Dirk Thatmann, Mathias Slawik, Sebastian Zickau, and Axel Küpper

Technische Universität Berlin
Service-centric Networking
{d.thatmann,mathias.slawik,sebastian.zickau,axel.kuepper}
@tu-berlin.de

Abstract. While cloud computing has seen widespread usage, there exist domains where the diminishing of management capabilities associated with cloud computing prevent adoption. One such domain is the health sector, which is the focus of the TRESOR[1] project. Enabling cloud computing usage under strict compliance constraints such as enterprise policies and legal regulations is the goal of TRESOR. The main approach consists of a distributed cloud proxy, acting as a trusted mediator between cloud consumers and service providers. In this paper we analyze issues which arise within the TRESOR context and show how an architecture for a proposed ecosystem bypasses these issues. The practicability of our solution is shown by a proof of concept proxy implementation. As all components of the architecture will be part of our proposed cloud ecosystem, we provide a holistic and generic proposal to regain management capabilities in cloud computing.

Keywords: Cloud Computing, Cloud Proxy, REST, SLA, Regulatory Compliance, Cloud Broker, Marketplace

1 Introduction

Cloud computing promises many advantages. Widely it is recognized as a viable way to reduce operational costs. These cost reductions are opposed by some pitfalls, for example, lack of convenience by missing features, no industry standards and therefore non-interoperable solutions, insufficient compliance to legal requirements, and missing security and privacy functionality resulting in untrustworthy relationships [29].

We have identified disadvantages and risks of cloud computing which are the main reasons for the hindered adoption of cloud computing within sensitive domains, such as the health sector. These disadvantages and risks can be summarized as follows:

Privacy, legal, and compliance issues. Most cloud computing solutions incorporate outsourcing over organizational and sometimes country borders.

[1] **TR**usted **E**cosystem for **S**tandardized and **O**pen cloud-based **R**esources

Within sensitive domains there are many guarantees, which have to be given regarding data privacy, legal compliance, and secure auditing. Some of them are reflected within acts, such as Payment Card Industry - Data Security Standards (PCI DSS), Sarbanes-Oxley (SOX) or Health Insurance Portability and Accountability Act (HIPAA). Special care has to be taken that the outsourcing provider fulfills these requirements [31] [18] [9]. Furthermore, hardware virtualization, storage abstraction, multi-tenancy, and container technologies allow flexible utility computing models, but sometimes introduce these issues themselves [5] [4] [31]. Also, laws and provisions are traditionally confined to national borders. As globally distributed cloud computing environments make these borders indistinct, the risk for enterprises not being compliant to these requirements is increasing.

Transparency. The inability to asses critical aspects, such as the mean time to repair (MTTR), is often caused by the fact that the cloud provider's contingency procedures, such as backup, restore or disaster recovery are not transparent to the cloud computing consumer. As with most other IT services, migrating to and using cloud computing services introduces follow-up costs, as shown in [20] and [17]. Some of these costs are hidden, for example, costs for making services compliant to regulations, backup, restore, and disaster recovery procedures.

High integration efforts. Within enterprise architectures, the means of integrating heterogeneous systems are manifold. Generalized, homogeneous, and invariable cloud computing services raise the integration effort of existing enterprise infrastructure, such as existing user databases or single sign on solutions, considerably.

Lock-in effects. Lock-in effects arise from the lack of industry standards enforcement and make migration to other providers difficult. The bankruptcy of a cloud service provider could have severe consequences if important enterprise services are hosted in the cloud.

Addressing cloud computing disadvantages and risks. To address these shortcomings, we propose a distributed cloud proxy for monitoring and controlling the cloud service consumption. This control is necessary to enable compliance to all privacy, legal, and regulatory issues regarding the service consumption. As the proxy is comparable to an application layer gateway for cloud computing services it reduces the integration effort, as it is able to integrate existing user databases for a manifold of different services. Common lock-in effects, such as dependencies on vendor tools or proprietary protocols are avoided, as the proxy will provide open APIs and is based on the widespread HTTP protocol applied within a REST-based architecture.

A cloud service description language formalizes aspects of cloud services on many levels, for example, technical interfaces, legal constraints, and business models. This description language will enhance the transparency of cloud services from different viewpoints like service-level agreements (SLAs), compliance, and

price models. Additionally, it is used by a cloud service broker to connect clients and providers of cloud services within a cloud service marketplace.

In the following chapter we show how the distributed proxy addresses theses issues. In chapter 3 we present a proof of concept implementation of the proxy and an assessment of our prototype. The paper concludes with a related work chapter and a summary and outlook.

2 The Cloud Proxy

The following subsections present some details of the cloud proxy. This includes its distribution and security, compliance and location-aware features, and its reliance on the REST architectural style. The last subsection explains, how the proxy relates to other proposed components of the cloud ecosystem.

2.1 The Cloud Proxy Distribution

The integration of existing enterprise systems, such as an Active Directory, into 3rd party services is easier and more secure if the communication is not extended to a public cloud environment. Furthermore, some management capabilities have to be realized by a 3rd party, independent from the cloud service client and the cloud service provider. This is especially true for the monitoring of SLAs: as pointed out by Koller et al. [21] an independent party can monitor and enforce SLAs more trustfully than the participating parties could do by themselves.

The management of cloud service consumption through a cloud proxy enables the service provider to rely on implicit guarantees - such as the correct client authentication or that all policies of service clients are met. Cloud services, which are accessed through the proxy, are released from the duty of implementing some AAA functionality, because the proxy can either locally authenticate users by using a simple password database or rely on existing single sign-on (SSO) solutions, such as Kerberos [25]. With all these factors in mind, we propose a distribution of the cloud proxy between the service client, a trusted, independent 3rd party, and the service provider.

In Figure 1 our distributed proxy and the functionality of the individual components are shown. The proxy is distributed between the service client organizations, a trusted party, and the service provider. All service consumption is managed by the distributed proxy and encrypted through a trusted cloud transfer protocol (see Section 2.4). The client proxy integrates local systems, such as user databases (e.g., LDAP, Active Directory). The trusted cloud proxy monitors and controls the connections from the client to 3rd party services.

2.2 Location-aware features

The cloud ecosystem includes novel approaches regarding location-aware cloud computing, which can be divided into four main categories:

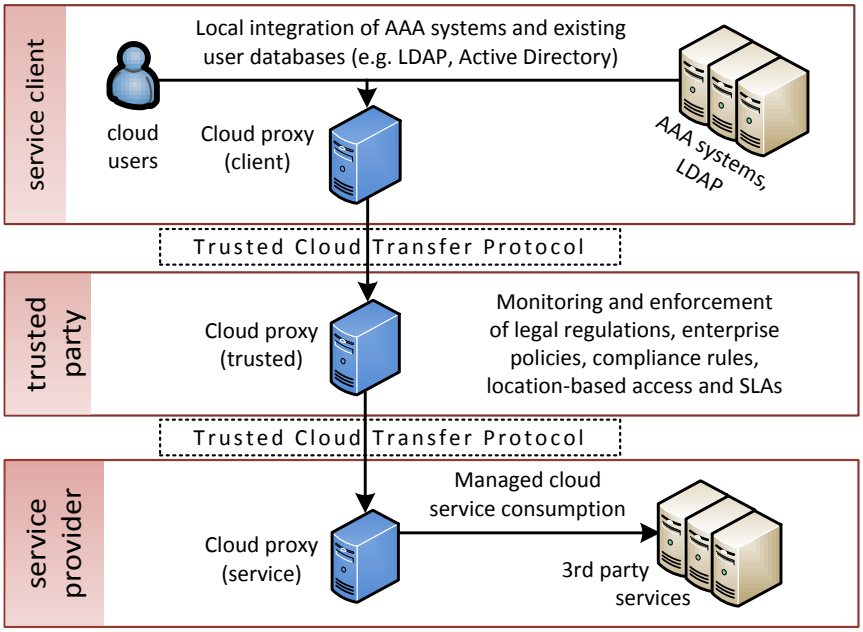


Fig. 1. The cloud proxy distribution.

Access control. We will reintroduce the former access restrictions of the physical boundary, for example, the enterprise premises, by enabling location-based access control for TRESOR services. Furthermore, we enable cloud service consumers to specify versatile policies to enable compliant and managed access to cloud services based on location information. This enhances existing concepts regarding location-based access control such as those proposed by Ardagna et al. [3]. The location information, which is used for enforcing location-based access control, can also be used by the cloud services to adapt their functionality based on the position of the consumers, thus realizing location-aware computing.

Compliance. As the proposed description language for TRESOR will include location information, e.g., the position of cloud computing resources, clients can assess the compliance of cloud services to regulations based on service locations such as EU data privacy laws.

Pricing. As traffic, maintenance, and hardware costs are not the same globally, the provisioning costs of cloud services may vary. To reflect these differing costs, the ecosystem allows the definition of different price models for service consumers based on countries or regions. As the ecosystem processes location information it can put these pricing models into practice.

2.3 Relying on REST

We identify the REST [16] architectural style as the prevalent[2] style for cloud service implementations. Many concepts of REST are suitable for using them to enable control over the data flow, for example the addressing of resources by URIs. The cloud proxy could easily match resource URIs with a set of patterns and connected access authorization rules. Unlike the SOAP/RPC-style, where each application may specify its own resource addressing scheme, this mechanism is applicable for all REST-based cloud services.

Meaningful REST URIs also enable operation and resource-based logging and accounting. Furthermore, HTTP includes information, which could be integrated into the SLA monitoring, for example, the HTTP status code. To make use of these advantages, all functional modules of the distributed proxy work with REST-based cloud services.

2.4 Trusted Cloud Transfer Protocol

The HTTP protocol separates the HTTP header, consisting of meta information about the request, from the HTTP body, which often contains sensitive application data [15]. To implement control and management functions within the distributed cloud proxy, only meta information about a request are needed and not the full message body.

Figure 2 shows the distributed cloud proxy, the connected systems, and their role within the Trusted Cloud Transfer Protocol (TCTP). All messages between the distributed cloud proxy instances are encrypted using TLS. To prevent the trusted party to access the content of all messages, we propose to use TLS not only for transport encryption, but additionally for encrypting the HTTP body between the service client and service provider. This enables direct control over the data flow without compromising end-to-end encryption of sensitive application data. As all out-of-band communication is prohibited, the trusted proxy and service proxy can rely on and trust the transmitted TRESOR identity.

2.5 Monitoring of SLA compliance

Most cloud providers only offer simple SLAs, such as the "Annual Uptime Percentage" SLA of Amazon EC2 [1]. Emeakaroha et al. propose enhanced cloud computing SLAs, such as *mean time to repair* (MTTR) or *mean time between failure* (MTBF), which is also an aspect of the proposed framework of Dobson et al [12]. As our proxy combines in-band and out-of-band information, it can provide similar SLAs in order to allow the definition of complex SLA requirements:

In-band information. The TCTP protocol enables access to the HTTP header of messages sent to the cloud proxy, which convey relevant SLA information, e.g., status codes, request URIs, user identities or location information.

[2] The largest directory of public cloud APIs identifies 70% of all 6.931 listed as being based on REST. [28]

Out-of-band information. In addition, some SLA information has to be gathered differently, i.e. through agents [13], plug-ins for external XaaS services or other APIs.

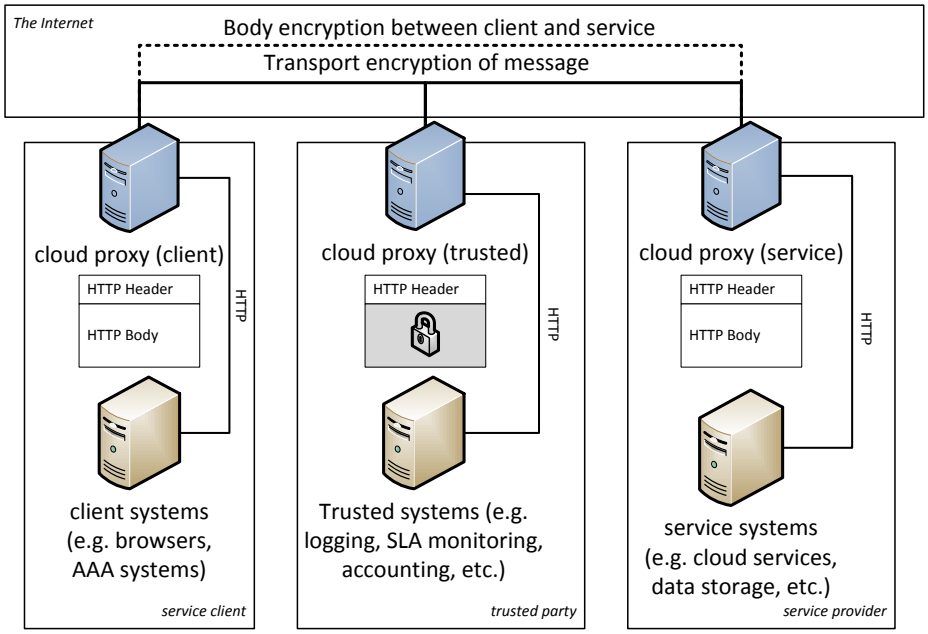


Fig. 2. The TCTP (Trusted Cloud Transfer Protocol) scheme

2.6 Further components of the cloud ecosystem

The TRESOR proxy is interconnected with many other components, which are are briefly described in the following paragraphs, as they are not in the focus of this paper.

The Cloud Service Description Language. The cloud service description language is a future aspect of the cloud ecosystem and will formalize technical, compliance, and business aspects. For the technical aspects, we will consider existing languages for inclusion, e.g., WSDL [10], USDL [26] or Linked-USDL [7]. Our formalization of SLAs will be based on the groundwork of ITIL's SLA definition. Besides the common areas, our focus lies on: Compliance with regulations by law, enterprise policy mappings, and enhanced network connection agreements. To enable service brokering and a marketplace, the cloud description language will incorporate business aspects, such as pricing information and payment models.

Service Broker and Marketplace. The proposed cloud service description language for TRESOR allows clients and providers to formalize their requirements and capabilities. The TRESOR broker then matches and suggests compatible cloud services based on these formalization. This automation considerably lowers the effort for clients to discover and select cloud services, which are compatible to the client requirements as this is now a manual and sometimes time consuming task.

Menychtas et al. [23] identify four major phases of electronic marketplaces. All processes within these areas are implemented and enriched by TRESOR components. The *information phase* is enhanced through the detailing within the cloud service description language. The description language also includes pricing information, which are the basis for the *negotiation and price setting phase*. The cloud broker matching result is cryptographically signed to form a legal agreement between service provider and consumer within the *Contracting phase*. As all communication is managed by the cloud proxy, it can implement independent metering functionality, which is used during the *settlement phase*.

3 Proof of concept

In the following, a proof of concept implementation of the cloud proxy is presented. On the basis of this implementation we make a preliminary analysis of the impact of our proposed cloud architecture.

3.1 Technology

The proxy uses non-blocking and asynchronous functions to enable highly scalable I/O operations using the Java New I/O (NIO) API [30]. To provide abstractions for the low-level functions of the Java New I/O API, we use the Grizzly Framework [19]. The Grizzly Framework has shown impressive performance characteristics, as shown in [24]. The Grizzly Framework also contains supporting implementations for processing HTTP packets and a customizable TLS engine, which assists us in the implementation of the Trusted Cloud Transfer Protocol (TCTP).

For modularization we rely on the industry standard OSGi [27]. As OSGi application server platform we chose Eclipse Virgo [32]. Eclipse Virgo eases the deployment effort, as the application OSGi modules, can be independently updated at runtime - a major requirement for central architectural components.

3.2 Architecture

The proof of concept architecture consists of two bundles: the *proxy model*, which contains a preliminary configuration, authentication, and an SLA model, and the *proxy core*, which reads a model instance and configures a proxy runtime object. For the proof of concept, we implemented the following functionality:

Authentication. The proxy matches URI patterns to authentication rules and authenticates users through a password database.

Relaying identities. After users are authenticated, the proxy relays their identities to the downstream proxies by using a special HTTP header. In the future, it could also relay roles to enable role-based access control (RBAC).

Routing and SSL. The proxies can encrypt traffic using SSL and route incoming messages as defined by the proxy model.

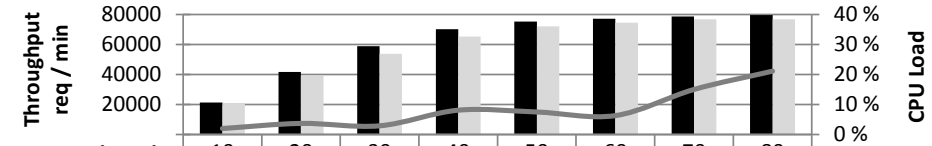
Monitoring of SLA compliance The Proxy monitors simple SLAs, e.g., logging application errors and comparing them to a defined maximum allowed percentage.

3.3 Evaluation

In this chapter, we evaluate the performance characteristics and the integration effort of the cloud proxy prototype.

Performance analysis. For performance analysis, the homepage of a simple Ruby on Rails web application is accessed through the Apache JMeter [2] load test tool. The application runs on a Linux server (Debian 6.0) with an Intel Core i7 930 CPU, 24 GByte RAM and is accessed using a HP EliteBook 8440p notebook PC (Core i7 620M, 8GByte RAM).

We compare the direct communication with the service with the communication through one instance of the proxy regarding the application throughput (requests per minute) and the CPU usage on the client computer to show the impact of the proxy processing on the application access. We varied the number of JMeter threads to simulate different parallel workloads. To cut out network impacts, the proxy is running on the machine used for accessing the service and furthermore, SSL is deactivated to exclude encryption overhead.



Threads	10	20	30	40	50	60	70	80
Direct	21354	41605	58877	70166	75184	76998	78522	79443
Proxy	20793	39106	53656	65146	71990	74419	76655	76718
Perf. Impact	- 3%	- 6%	- 9%	- 7%	- 4%	- 3%	- 2%	- 3%
CPU Load	2 %	4 %	3 %	8 %	8 %	6 %	15 %	21 %

Fig. 3. Impact of the cloud proxy prototype on the application throughput

Our results are shown in Figure 3. We see that the Proxy impacts the throughput 9% at most. The server CPU starts to saturate when using 50 parallel

threads. The overall application throughput does not increase significantly if the number of threads is increased. At this stage of the implementation we see that the chosen technology does not impact the overall performance of the proxy in a substantial way.

Integrating the proxy authentication. We modified a sample Ruby on Rails application to use the relayed identity of a service user to analyze how the proxy authentication could be integrated into existing cloud services. Our evaluation shows that it is very easy to modify such a contemporary RESTful web application to use the supplied proxy authentication information. If this holds true for other web frameworks, this mechanism could therefore lead to reduced implementation efforts for proxy-compatible applications.

4 Related work

The approach of a multi-role distributed proxy is in line with the idea of Weissman et al. [33] which states that "enabling proxies to assume multiple roles is key to the performance and reliability of distributed data-intensive multi-cloud applications". In order to follow this idea a representation of SLAs is needed, which the distributed cloud proxy will be enforcing. A number of such representations is available, for example, OWL ([11], [12]), WSLA ([6], [14]), and RDF ([8] and [22]).

Many works depict cloud proxies with additional roles, for example, mitigating constraints of mobile devices [35] or realizing a certificate-less re-encryption scheme [34].

5 Summary and Outlook

In this paper we have presented a distributed cloud proxy with the goal of regaining management capabilities within cloud computing environments. As compliance rules will be formalized through a cloud service description language, the cloud proxy allows compliant and managed cloud service consumption. This enables novel cloud computing services within sensitive domains with many compliance regulations, such as the health sector. The solution is also considerably more secure as no authentication information is sent to any system outside of the client organization. The preliminary analysis of the cloud proxy prototype shows the applicability of our approach regarding performance characteristics and integration effort. Future work will include the extension of the cloud proxy, implementing the other components of the cloud ecosystem, and the provisioning of initial services in collaboration with associated project partners from health care institutions.

Acknowledgement. The work presented in this paper was performed in the context of the TRESOR project. TRESOR is funded by the German Federal Ministry of Economics and Technology (BMWi).

References

1. Amazon Web Services LLC: Amazon EC2 SLA. http://aws.amazon.com/en/ec2-sla/ (2008)
2. Apache Software Foundation: Apache JMeter. http://jmeter.apache.org/ (2012)
3. Ardagna, C.A., Cremonini, M., di Vimercati, S.D.C., Samarati, P.: Access Control in Location-Based Services. Privacy in Location-Based Applications Privacy in Location-Based Applications, LNCS 5599, 106–126 (2009)
4. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M.: A view of cloud computing. Commun. ACM 53(4), 50–58 (Apr 2010), http://doi.acm.org/10.1145/1721654.1721672
5. Brandic, I., Dustdar, S., Anstett, T., Schumm, D., Leymann, F., Konrad, R.: Compliant Cloud Computing (C3): Architecture and Language Support for User-Driven Compliance Management in Clouds. In: Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on. pp. 244 –251 (july 2010)
6. Brandic, I., Music, D., Leitner, P., Dustdar, S.: VieSLAF Framework: Enabling Adaptive and Versatile SLA-Management. In: Altmann, J., Buyya, R., Rana, O. (eds.) Grid Economics and Business Models, Lecture Notes in Computer Science, vol. 5745, pp. 60–73. Springer Berlin / Heidelberg (2009), http://dx.doi.org/10.1007/978-3-642-03864-8_5
7. Carlos Pedrinaci, T.L.: Linked USDL. http://www.linked-usdl.org/ (2012), http://www.linked-usdl.org/
8. Chaudhary, T.C.S., Kumar, V., Bhise, M.: Service Level Agreement parameter matching in Cloud Computing. In: Proceedings of the World Congress on Information and Communication Technologies 2011. IEEE (2011)
9. Chen, L., Hoang, D.: Novel Data Protection Model in Healthcare Cloud. In: High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on. pp. 550 –555 (sept 2011)
10. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S.: Web Services Description Language. http://www.w3.org/TR/wsdl (2001), http://www.w3.org/TR/wsdl
11. Dobson, G., Lock, R., Sommerville, I.: QoSOnt: a QoS ontology for service-centric systems. In: 31st EUROMICRO Conference on Software Engineering and Advanced Applications. pp. 80 – 87 (sept 2005)
12. Dobson, G., Sánchez-Macián, A.: Towards unified QoS/SLA ontologies. In: Proceedings of Third International Workshop on Semantic and Dynamic Web Processes (SDWP 2006) (2006)
13. Emeakaroha, V.C., Brandic, I., Maurer, M., Dustdar, S.: Low level Metrics to High level SLAs - LoM2HiS framework: Bridging the gap between monitored metrics and SLA parameters in cloud environments. In: High Performance Computing and Simulation (HPCS), 2010 International Conference on. pp. 48 –54 (28 2010-july 2 2010)
14. Emeakaroha, V.C., Ferreto, T.C., Netto, M.A.S., Brandic, I., Rose, C.A.D.: CASViD: Application Level Monitoring for SLA Violation Detection in Clouds. In: Proceedings of the 36th Annual IEEE Computer and Application International Conference (COMPSAC'12). Izmir, Turkey (2012)
15. Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T.: Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard) (Jun 1999), http://www.ietf.org/rfc/rfc2616.txt, updated by RFCs 2817, 5785, 6266, 6585

16. Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine (2000), 2000
17. Gmach, D., Rolia, J., Cherkasova, L.: Comparing efficiency and costs of cloud computing models. In: Network Operations and Management Symposium (NOMS), 2012 IEEE. pp. 647 –650 (april 2012)
18. Gonzalez, R., Gasco, J., Llopis, J.: Information Systems Outsourcing Reasons and Risks: An Empirical Study. Industrial Management and Data Systems 110(2), 284–303 (2009)
19. java.net: index.html - Java.net. http://grizzly.java.net/ (2012)
20. Kashef, M.M., Altmann, J.: A cost model for hybrid clouds. In: Proceedings of the 8th international conference on Economics of Grids, Clouds, Systems, and Services. pp. 46–60. GECON'11, Springer-Verlag, Berlin, Heidelberg (2011), http://dx.doi.org/10.1007/978-3-642-28675-9_4
21. Koller, B., Schubert, L.: Towards autonomous SLA management using a proxy-like approach. Multiagent Grid Syst. 3(3), 313–325 (Aug 2007), http://dl.acm.org/citation.cfm?id=1375627.1375631
22. Leidig, T., Momm, C.: USDL Service Level Agreements. http://www.linked-usdl.org/ns/usdl-sla (April 2012)
23. Menychtas, A., Gomez, S.G., Giessmann, A., Gatzioura, A., Stanoevska, K., Vogel, J., Moulos, V.: A Marketplace Framework for Trading Cloud-Based Services. GECON 2011 LNCS 7150, 77–89 (2011)
24. Mkrtchyan, T.: dCache: implementing a high-end NFSv4.1 service using a Java NIO framework. Computing in High Energy and Nuclear Physics (CHEP) (2012)
25. Neuman, C., Yu, T., Hartman, S., Raeburn, K.: The Kerberos Network Authentication Service (V5). RFC 4120 (Proposed Standard) (Jul 2005), http://www.ietf.org/rfc/rfc4120.txt, updated by RFCs 4537, 5021, 5896, 6111, 6112, 6113, 6649
26. Oberle, D., Barros, A., Kylau, U., Heinzl, S.: A unified description language for human to automated services (2012), http://dx.doi.org/10.1016/j.is.2012.06.004, in press
27. OSGi Alliance: OSGi Alliance — Main / OSGi Alliance. http://www.osgi.org/Main/HomePage (2012)
28. Programmable Web: Protocol usage by APIs. http://www.programmableweb.com/images/charts/TopProtocolsAlltime.png (2012)
29. Qi Zhang, Lu Cheng, R.B.: Cloud computing: state-of-the-art and research challenges. Journal of Internet Services and Applications 1, 7–18 (Mai 2010)
30. Reinhold, M.: New I/O APIs for the JavaTM Platform. http://www.jcp.org/en/jsr/detail?id=51 (2002)
31. Sengupta, S., Kaulgud, V., Sharma, V.: Cloud Computing Security–Trends and Research Directions. In: Services (SERVICES), 2011 IEEE World Congress on. pp. 524 –531 (july 2011)
32. The Eclipse Foundation: Virgo - Home. http://www.eclipse.org/virgo/ (2012)
33. Weissman, J., Ramakrishnan, S.: Using Proxies to Accelerate Cloud Applications. In: Proceedings of HotCloud 09 - Workshop on Hot Topics in Cloud Computing (2009)
34. Wu, X., Xu, L., Zhang, X.: Poster: a certificateless proxy re-encryption scheme for cloud-based data sharing. In: Proceedings of the 18th ACM conference on Computer and communications security. pp. 869–872. CCS '11, ACM, New York, NY, USA (2011), http://doi.acm.org/10.1145/2093476.2093514
35. Zhu, W., Luo, C., Wang, J., Li, S.: Multimedia Cloud Computing. Signal Processing Magazine, IEEE 28(3), 59 –69 (may 2011)