

Supporting end-to-end resource virtualization for Web 2.0 applications using Service Oriented Architecture

C. Papagianni¹, G. Karagiannis², N. D. Tselikas³, E. Sfakianakis⁴, I.P. Chochliouros⁴, D. Kabilafkas⁵, T. Cinkler⁶, L. Westberg⁷, P. Sjödin⁸, M. Hidell⁸, S. Heemstra de Groot⁹, T. Kontos¹⁰, C.Katsigiannis¹, C. Pappas¹, A. Antonakopoulou¹, I. S. Venieris¹

¹ School of Electrical and Computer Engineering, National Technical University of Athens, Greece

² CTIT and Department of Electrical Engineering, Mathematics and Computer Science
University of Twente, Enschede, the Netherlands

³ Department of Telecommunications Science and Technology, University of Peloponnese, Greece

⁴ Labs & New Technologies Division, Research Programs Section,
Hellenic Telecommunications Organization S.A.

⁵ Labs & New Technologies Division, Switching & Services Laboratory,
Hellenic Telecommunications Organization S.A.

⁶ Department of Telecommunications and Media Informatics,
Budapest University of Technology and Economics, Hungary

⁷ Ericsson Research, Stockholm, Sweden

⁸ School of Information and Communication Technology, Royal Institute of Technology, Stockholm, Sweden

⁹ Twente Institute for Wireless and Mobile Communication, Enschede, the Netherlands

¹⁰ Siemens IT Solutions and Services, Athens, Greece

Abstract— In recent years, technologies have been introduced offering a large amount of computing and networking resources. New applications such as Google AdSense and BitTorrent can profit from the use of these resources. An efficient way of discovering and reserving these resources is by using the Service Oriented Architecture (SOA) concept. SOA can be considered as a philosophy or paradigm in organizing and utilizing services and capabilities that may be under the control of different ownership domains. This paper presents an architecture that can be used to support end-to-end resource virtualization for Web 2.0 applications and in particular for peer-to-peer applications by using the Service Oriented Architecture concept.

Keywords— Web 2.0, SOA, P2P, resource virtualization

I. INTRODUCTION

THERE currently exist new means of online communication and collaboration, as well as new ways of producing and consuming information on the web [1]. These new ways are often denoted as Web 2.0 applications [2]. Examples of such applications are Google AdSense, Flickr, Wikipedia, web services, YouTube etc. In addition, peer-to-peer (P2P) applications such as BitTorrent impose further requirements on the underlying network infrastructure and on the available resources.

Furthermore, computing and communication are being blended into new ways of using networked computing systems. The network retains its traditional role as a means of

information exchange but, at the same time, it is also perceived as a means of sharing resources, where communication networks are used to share programs, information, processing capacity, communication, storage, and media. Next generation network and service infrastructures should overcome the scalability, flexibility, resilience, mobility and security bottlenecks of current network and service architectures, in order to provide a large variety of business models capable of dynamic and seamless end-to-end utilization of shared resources across a multiplicity of devices, networks, providers and service domains.

The concept of virtualization constitutes a promising one for addressing the requirements for the future of the Internet. With virtualization, multiple virtual networks are established over a shared, physical infrastructure, and each virtual network appears to have its own unique set of properties and performance characteristics. Hence, virtualisation inherently offers many qualities that are essential for the network of the future: isolation of traffic into virtual networks maintains privacy; decoupling of physical resources from network topology allows for resiliency and redundancy; confinement of service requirements within virtual networks makes it possible to address complexity and scalability.

Virtualization exists in the context of computing as well as in networks, but current virtualization techniques suffer from being based on managed configurations with static properties. This makes management and service provisioning cumbersome

and significantly limits the areas of applicability: In computing, virtualization is mainly used for the sharing of resources in restricted user communities such as Grid networks [3] and server farms, while in networking, it is primarily used for virtual private network services.

The Service Oriented Architecture (SOA) can be considered as a philosophy or paradigm to organize and utilize services and capabilities that may be under the control of different ownership domains, see [4]. SOA can also be seen as a way of promoting reuse, growth and interoperability by enabling users and organizations to get more value from capabilities, such as local resources as well as external ones.

The main goal of this paper is to present an architecture that can support end-to-end resource virtualization for Web 2.0 applications, P2P in particular, by using the SOA concept. The research questions answered by this paper are:

- 1) What are the requirements imposed by Web 2.0 applications and in particular by P2P applications?
- 2) Is there a benefit to providing end-to-end resource virtualization?
- 3) Which architecture(s) can be used to support end-to-end resource virtualization for Web 2.0 applications and in particular P2P applications?

The remaining part of this paper is organized as follows: In Section II concepts used in this paper are described in a more detailed way. The requirements imposed by Web 2.0 and P2P applications are discussed in Section III. Section IV describes a proposed architecture that can be used to support the requirements described in Section III. Conclusions and possible future work activities are described in Section V.

II. CONCEPTS USED

A. Web 2.0 applications

Different definitions are given to the term “Web 2.0”. For example, Tim O’Reilly, who has popularized Web 2.0, defined Web 2.0 [2] as a platform which is far more than a collection of web sites, which incorporates many new advanced applications, such as: Google AdSense, Flickr, Napster, BitTorrent, wikis, YouTube, Google Maps. In particular, [2] emphasizes that P2P applications, such as BitTorrent, demonstrate a key Web 2.0 principle, i.e., the service automatically gets better the more people will use it.

In [1] the following definition is given for Web 2.0: “Web 2.0 is defined as the philosophy of mutually maximizing collective intelligence and added value for each participant by formalized and dynamic information sharing and creation”. According to [1] and [5] the following features are associated with Web 2.0 applications:

- Support of communities that are aiming to unify users by using common ideals such as social networking or knowledge sharing.
- Platforms and tools that help users create, manage and maintain shared content with a broad audience.

- Support of online collaboration tools that can be used by users in collaboratively performing certain tasks.
- Use of the Representational State Transfer (REST) [6] to enable web clients to interact with arbitrary web resources in a uniform way.
- Use of the Really Simple Syndication (RSS) format to easily aggregate content from arbitrary sources.
- Use of AJAX which is defined as a combination of the Asynchronous JavaScript and XML and allows for rich user experiences (e.g. in Google Maps,).
- Support customers that are making use of Web 2.0, which can potentially be every web user.

B. Service-Oriented Architecture (SOA)

According to the Organization for the Advancement of Structured Information Standards (OASIS), SOA can be defined as “*a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with, and use capabilities to produce desired effects consistent with measurable preconditions and expectations*” [4].

According to the SOA model, a service provider publishes his service interface via a service registry where a service request can find the service and, subsequently, may bind to the service provider. Services can be accessed using well defined service interfaces and exercised following a service contract with certain policies. This provides a loose coupling of services and ensures operational agility. According to [4] and [5] the following features are associated with SOA:

- Allowance of a cross-organizational integration of services, by using common standards for the description of service interfaces.
- Facilitation of the inter-organizational integration of disparate services. This can be accomplished by using a central integration layer where heterogeneous applications can be encapsulated, and seamlessly integrated into an information technology landscape.
- Significant reduction of the development time, thanks to the availability of reusable application building blocks.
- Use of the Web Service Description Language (WSDL) for uniform service interface descriptions [7].
- Use of Web services based on Simple Object Access Protocol (SOAP) [8] where, among others, a protocol is specified for message exchange between services.
- Use of the Universal Description Discovery and Integration (UDDI) standard that allows for open service discovery.
- Support of customers that are making use of SOA, which are medium-sized or larger corporations.

C. Service architectures for shared computing resources

Several different computational models have been proposed to run on top of virtual machines across the Internet, such as Grid computing, on-demand computing, utility computing, and virtual desktop computing. Grid computing [9] aims at solving massive computational problems by using a large amount of virtualized computing resources in a network. On-demand computing and utility computing are related to making computing resources available across a network on a per-usage basis, with the goal to allow for cost-efficient utilization and sharing of resources. Virtual desktop computing is primarily oriented towards network-transparent and location-independent access to resources such as operating systems, displays, and networks [10] [11].

The aforementioned computational models address different aspects of resource sharing. A key issue for Grid systems is resource discovery to locate hardware and software resources for computational-oriented applications [12]. The purpose of utility computing is to make computing resources available as reliable commodity services, by employing a more efficient use of resources through dynamic and intelligent reconfiguration on the fly. Virtual desktop computing focuses on the actual methods for virtualization of different types of resources and the accessing of virtualized resources.

D. Service architectures based on peer-to-peer resource sharing

The basic principle of a P2P system is the sharing of resources (e.g., storage, bandwidth) among a large community of users. The P2P model differs from the traditional client-server model in the sense that participating peer nodes act as both clients and servers simultaneously. File sharing applications (e.g., Napster [13], Gnutella [14], Kazaa [15]) were among the first to exploit the pure hybrid P2P networking concept. The P2P concept of resource sharing has been adapted by other types of applications, such as for media streaming (e.g., CoolStreaming [16], GridMedia [16]) and distribution of large files to a large community of users (BitTorrent [18]). One of the limitations of the first P2P systems was related to centralized storage, leading to scalability and reliability problems. These issues have been addressed by using a Distributed Hash Table (DHT) concept, e.g. [19].

E. Network resource virtualization and control

Virtualization has become an important networking component for interconnecting groups of network nodes into virtual private networks (VPN), e.g. [20], for instance over a common MPLS/GMPLS network. In this context, virtual private networks are provisioned as LSP (Label Switched Paths) tunnels across a shared switched infrastructure, mainly based on manual static configurations. Current access and control methods for virtual networking are based on statistical multiplexing of simple conceptual models such as virtual links (L2TP, VPWS), virtual switches (VPLS) or virtual routers (L3VPN). These simplistic models have the advantage of keeping the complexity and the details of the underlying infrastructure invisible (and inaccessible) to customers, and instead provide them with a uniform, simple abstraction as the interface. However, there is little flexibility in the service

provided to the end-user who has no means of controlling or monitoring network resources. NFS's GENI initiative [21] uses a more refined network concept where resources (links, forwarders, storage, processor clusters) are *substrates*, and virtualization stands as an abstraction level to hide the underlying physical resources that can be shared by multiple infrastructures. CANARIE has developed the User Controlled Light Path (UCLP) [22], which enables users to define their own network architectures including topology, routing, virtual routers, switches, etc., by using web services techniques.

III. PEER-TO-PEER REQUIREMENTS ON END-TO-END RESOURCE VIRTUALIZATION

This section discusses the P2P requirements that have to be supported by the end-to-end resource virtualization solution.

A. Requirements

The P2P requirements presented below are partially based on the requirements given in [23].

1) *A distributed service architecture allowing for network-wide control and management of shared resources.*

Currently P2P services suffer from not being able to allocate dedicated services to their overlay networks. P2P users have no or little control of the network resources on top of which the overlay network runs, which means that P2P systems have to rely entirely on application-level control to achieve the desired networking properties. Operators and ISPs, on the other hand, have limited possibilities of providing value-added services for P2P applications. The service architecture should enable end-users to provide services, based on network-wide resource sharing.

2) *Transparent resource provisioning, for facilitating effortless end-user control over virtual resources.*

Control mechanisms and policies may vary significantly between different P2P user communities, service models, applications etc; this requires a wide variety of different control policies and transparency in resource provisioning. It also necessitates the decoupling of resource control from resource ownership. Models and abstractions for effortless and novel end-user control and management of virtualized resources are needed.

3) *Scalability.*

A great advantage of P2P systems offering seemingly boundless resources is their outreach to a vast number of users. However, this also includes the administration of an equivalent number of different domains. This necessitates further research efforts and the proper designing of various decentralized architectures that will be able to incorporate vast numbers of users and systems seamlessly, and at the same time continuously provide a wide variety of services.

4) *Connectivity*

For P2P systems to work efficiently end-to-end transparency is required; but the technologies existing at the network edge may differ greatly in terms of capabilities and throughput, thus hindering the ability to provide and control a

service in a ubiquitous and uniform way. This applies both to home and enterprise environments.

5) *Dynamic and distributed discovery*

Decentralized and distributed systems such as P2P often rely upon various discovery mechanisms in order to be able to recognize their peers and distinguish the services that are available to them. This becomes fairly evident, considering the fact that most peers of such systems reside in the Internet space with very limited transparency and knowledge of the intermediate network that provides connectivity between them.

6) *Security*

The concept of security in an architecture that can be used to support resource virtualization for P2P applications is difficult to address. Centralized environments tend to be fairly isolated with their important units secluded and managed by specialists. P2P systems, on the other hand, tend to be open and rely on different mechanisms focusing on user privacy, anonymity and trust. Such mechanisms often include community-based trust, replication and verification.

7) *Resource availability and failure management*

In a typical P2P system 50% of its registered resources may be unavailable on a 24 hour cycle. Because of the dynamic nature of such systems, resource availability and additional fail-stop models are necessary in order to provide seamless and ubiquitous service availability.

8) *Location Awareness*

Location awareness is the ability of an application to adapt its functionality according to information on geographical positioning. In the case of grid and P2P systems this information is significant, because apart from providing location-based services to users, it can also facilitate network and system level optimizations. It can prevent peers from randomly choosing logical neighbors, thus avoiding causing serious topology mismatches between the P2P overlay network and the physical underlying network. Additionally, location awareness can help to choose the shortest routes for data exchange, instead of using flooding-based search mechanisms.

9) *Group Support*

As multitasking or collaborative systems, P2P needs to employ mechanisms for creating and managing groups easily. Such groups may include sharing of multimedia content on an occasional basis, such as in social networks where one can share photos, music, etc.

B. *Example*

Virtual content distribution networks for video distribution are utilized as an example to illustrate how services can be established with peer-to-peer techniques using virtualized resources in end-user clients, servers and network infrastructures. In a content distribution network, content is replicated over the infrastructure to multiple locations, and client requests are redirected to the most suitable location based on network-related criteria such as traffic volume, quality, and proximity, as well as criteria related to content storage such as server load, response time, and availability.

Fig. 1 shows a content distribution scenario with multiple content distribution networks on top of the same physical resources. There are three types of actors, where *end-users* subscribe to content services, and at the same time make storage and processing resources available in a peer-to-peer manner. *Resource service providers* specialize in content distribution resource provisioning, and can provide virtualised resources for application-specific purposes such as video head-end functionality, backup and recovery, etc. as well as for content storing and processing. *Content providers* utilize the provided resources for purposes of replicating content to multiple locations, for routing client requests to content locations, for meeting performance guarantees in delivering contents to clients, and so on. The content provider retains the full administrative control of the virtualised resources.

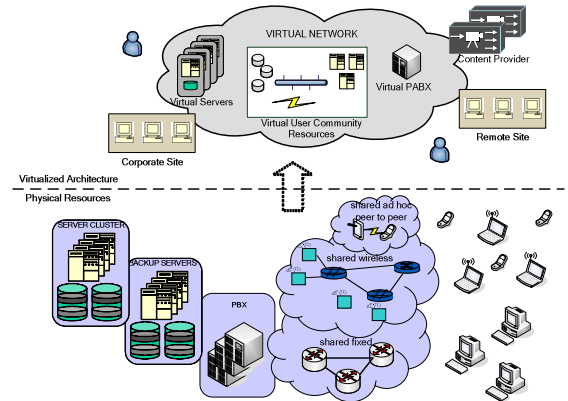


Figure 1. Virtual content distribution network with P2P techniques

In this way the distribution network is transformed from a centralized, hierarchical and possibly cost-ineffective structure to a peer-to-peer-like distributed application where the required physical elements become virtualized. The decoupling of content provisioning from resource ownership makes it possible to realize content sharing services without direct investments in infrastructure, by taking advantage of the ubiquity of resources available through end-user equipment.

IV. SYSTEM FUNCTIONAL DESCRIPTION AND ARCHITECTURE

This section describes an architecture that is used to support the requirements presented in Section III. To the knowledge of the authors of this paper, the only existing architecture that can support a subset of these requirements is the Open Grid Services Architecture (OGSA) [3]. Currently, OGSA can support grid-based resource virtualization. In particular, the P2P requirements in Section III are partially based on [23].

Existing models for resource sharing suffer from several limitations on supporting end-to-end resource virtualization. Grids have a strong focus on computational resources only, while utility computing is server-oriented. Furthermore, the control and management schemes for these computational models are, to a large extent, based on managed configurations with static properties. In addition, even though shared resources are virtualized and made available for network-wide

use, management and control of the resources are still retained within the context of the resource provider.

The goal of the architecture presented here is to bring about advances in all these areas by applying SOA and P2P principles, by which resource discovery mechanisms could be explored from the starting point of a decentralized semantic-based P2P resource discovery service ([24], [25]). Also, interoperability among various administrative domains is a fundamental property. Problems imposed by the server-centric and static nature of resource sharing in utility computing may be effectively addressed by the proposed architecture, which adopts unified resource sharing as its main design principle. By layering virtualization over existing network topologies, associated computing resources as well as services will provide an effective solution to this problem. Finally, the proposed architecture enables third-party control and management of the resource sharing process, as well as of the shared resources themselves.

The architecture proposes mechanisms for decoupling resource control from resource ownership to enable the development of new control policies. This relies on the separation of the control and forwarding plane functionalities in network systems, to realize control functionality in a manner independent of the forwarding plane. Hence, control and management decisions can be made within the virtualized networks. Architectures with separation of routing and forwarding have been proposed to address control and management issues for the Internet ([26], [27]). We extend this concept to the context of resource sharing in virtual networks, to make it possible to treat switching devices (e.g., routers, switches and cross-connects) as networking resources subject to virtualization.

The architecture is designed to enable P2P models for resource sharing where communities of users can cooperate to make efficient use of computing and communication resources across the network. Furthermore, the introduction of unified management of virtual resources is set to enable operators to provide network resources tailored to P2P user communities; this will improve the quality of P2P network services, and will generate operator revenues from these services.

The architecture presented in this section reuses the SOA and OGSA principles. It is important to note that the intention of the authors of this paper is to provide insights on end-to-end virtualization support that could be subsequently reused and applied within OGSA.

Fig. 2 illustrates explicitly the proposed architecture. Each layer is comprised of several sub-components that provide the functionality of the layer. Modules at the top layer provide services such as controlling virtualization, protecting privacy, discovering and scheduling resources, trading and optimization to the application layer through the application interface.

These services are built upon the lower layer functional components. The functional components located in one layer can share common characteristics and can build on the capabilities provided by the lower layers. The system layers interact with each other by means of bottom-up activation and top-down execution. Due to the paper page length limitation,

the remaining part of this section is focusing only on the Unified Resource Virtualization, Control and Support tier. An example on how the Shared Resources Tier can be realized can be found in [3].

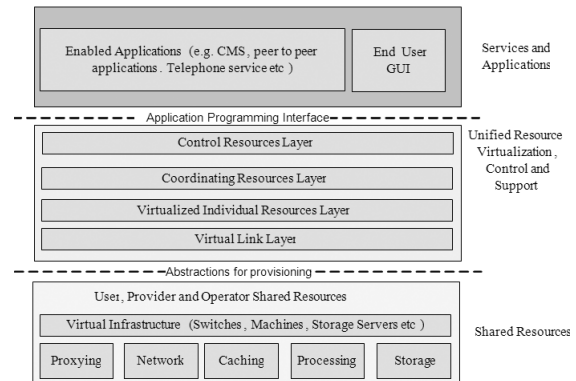


Figure 2. Conceptual Architecture

Fig. 3 suggests the implementation of a Unified Resource Virtualization, Control and Support tier. Through the proposed middleware, uniform and location-independent methods for access to and control of virtualized resources are provided, independently of the specific properties of the underlying resources. This tier establishes secure connectivity to the underlying shared physical resources through the Virtual Link Layer; registers, manages and controls resources through the Virtualized Resources Layer; handles interactions across collections or pools of resources through the Coordinating Resources Layer, and; performs resource discovery and scheduling, as well as trading and optimization of resource usage, through the Control Resources Layer.

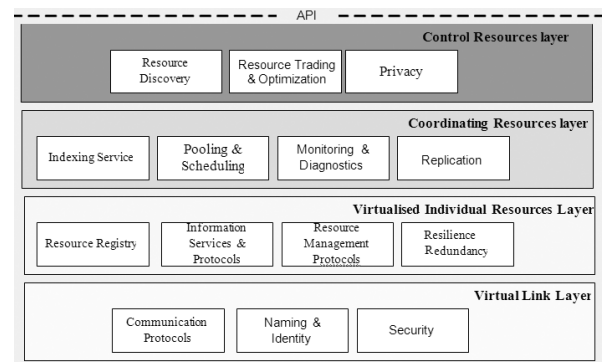


Figure 3. Unified resource virtualization, control and support tier

A. Virtual Link Layer (L1)

The Virtual Link Layer consists of functional components required to provide a secure exchange of data between the shared physical resources.

Communication Protocols provide the means for all components to exchange information. The set of candidate technologies/mechanism include RPC, Web Services, classical middleware like CORBA, P2P middleware like JXTA, tuple spaces middleware as well as future SIP P2P from IETF.

The **Naming and Identity** component guarantees unique entity names. Entities could be associated with a human-readable name, location and space independent name and location address such as an IP address/port pair.

Sharing of resources requires access control through robust **security** mechanisms and protocols. This can be accomplished by using authentication and authorization mechanisms in order to identify the user and enforce associated policies on resources, isolation of user identity, performance and available content, delegation of access rights, security policy exchange and intrusion detection.

B. Virtualized Individual Resources Layer (L2)

The Virtualized Individual Resources Layer builds on the connectivity layer to provide mechanisms and protocols for the secure negotiation, resource management and control of sharing operations on the individual shared resources. The functionality components that can be included in this layer are the resource registry, information services and protocols, resource management protocols, resilience and redundancy.

A **Resource Registry** has to be maintained that includes the structure and state of the resource, for example its name and configuration, the owner of the resource, the current load and usage policy.

Information Services and Protocols are used to communicate with the shared physical resources in order to obtain the information concerning the shared resources.

Resource Management Protocols are used to negotiate access to a shared resource, and to provide control over the resource sharing operation (e.g., termination, and monitoring support over the status of the resource sharing operation).

Resilience and Redundancy can be provided by realising fault tolerant hardware and software systems. Fault management mechanisms ensure that running processes and jobs are not lost due to resource faults. Moreover, mechanisms are required to provide for replacement/redundancy on disconnection of physical resources.

C. Coordinating Resources Layer (L3)

The Coordinating Resources Layer builds on the individual resource layer and contains mechanisms and protocols that are global in nature and support interactions across collections or pools of resources. Such functionality components are the indexing service, pooling and scheduling service, monitoring/diagnostics service and content replication.

An **Indexing Service** has to be maintained that holds/retrieves information about a set of shared resources registered to it. Primarily, it provides an interface for enabling queries concerning registered resources.

Pooling and Scheduling services allow users to allocate or pool virtualized resources for a specific purpose, and for the scheduling of different tasks on the appropriate resources.

Monitoring and Diagnostics services allow the monitoring of the virtualized resources for loss of functionality, intrusion detection, overload, etc.

Content Replication supports management of data and content storage, in order to maximise access performance in terms of response time, reliability, redundancy, cost, etc.

D. Control Resources Layer (L4)

The Control Resources Layer is the last layer to the end-user; therefore it must provide all the additional functionalities that grant this sophisticated service architecture, far from being just a plain unrefined system for sharing resources. The functionality components that are included in this layer are the Resource Discovery and Scheduling module, the Privacy module and the Trading and Optimization Support module.

The **Resource Discovery and Scheduling** module is responsible for the discovery of the resources that will be made available to the end-user. In cooperation with the Indexing Service in L3 and a service registry (e.g. UDDI), the Resource Discovery and Scheduling module searches for available virtual resources in order to serve user demands, invoking subsequently the appropriate trading negotiation mechanism supported by the Trading and Optimization Support module. Semantic matching may be supported through the use of an ontology repository. Similarly to Grid computing, the module will also request the execution of a job and allocation/aggregation of resources, in cooperation with the Pooling and Scheduling module in L3.

The purpose of the **Privacy** module is to establish a privacy framework that will protect end-user privacy, data confidentiality and integrity. Possible dangers concerning end-users' privacy may be the disclosure of their personal data and the corresponding tracking of resource usage patterns, as well as the confidentiality and integrity of data stored (or commuted) through virtual resources. A threefold approach includes a privacy-enhanced identity management system to deal with end-user registration on the service architecture, access control to shared/traded resources with privacy extensions and distributed dispersal of information enforcement (e.g. storage). The Privacy module will interact with the Security module in L1, which is actually the security enforcing point, and the Resource Registry, which will maintain the corresponding information for the virtual resource.

The **Trading and Optimization Support** module acts as a "resource trading broker", enabling indirect access to physical resources in accordance to the various business models supported by the service architecture. This broker will undertake the negotiation actions between the end-user and the owner of the resource. The critical activity here is to identify and choose the proper negotiation mechanism so that the user will participate in the trading process only in the case where particular quality guaranties for the resource are needed, or the resource is not available on a best-effort base.

The proposed architecture is set out to reconcile opposite directions on addressing security issues and user privacy. A vertical approach is followed where all tiers in the layered architecture deal with complementary aspects of security and privacy. Access control, security policy exchange and survivability concerning active network attacks on the shared application and networked resources are handled by the

Security module (L1); identity management, privacy and anonymity are handled by the Privacy component (L4); data confidentiality and integrity are vertically addressed in L1 (Security), L2 (Resource Registry), L3 (Replication) and L4 (Privacy); connectivity requirements are addressed in the context of robust unique entity naming by L1 (Naming and Identity and Communication Protocols); resource availability and failure management requirements are met in L2 (Resource Management Protocols and Resilience/Redundancy); finally monitoring and diagnostics services are provided by L3 (Monitoring and Diagnostics) to L4 and the application layer.

The architecture utilizes a decentralized P2P resource discovery service located in L4 (Resource Discovery module) in cooperation with the Indexing Service, the Pooling and Scheduling module and the Monitoring and Diagnostics service in L3. Provisioning of added value resource sharing is enabled through means of resource trading in L4 (Resource Trading and Optimization module) and decoupling resource control from resource ownership in L1/L2.

V. CONCLUSIONS AND FUTURE WORK

This paper presents an architecture that can be used to support end-to-end resource virtualization for Web 2.0 applications and in particular P2P applications, by using the SOA concept.

Taking into account the advances coming from existing resource sharing architectures (e.g. grid computing), the proposed architecture is designed to enable P2P models for resource sharing where communities of users will cooperate to make efficient use of computing and communication resources. The introduction of unified management of remote resources will enable resource providers to provide virtual resources tailored to P2P user communities. In addition, trading of virtualized resources brings forth a new service architecture where resource ownership and administrative rights can be traded between end-users. Future activities will mainly focus on the implementation and evaluation of the architecture presented in this paper.

ACKNOWLEDGEMENTS

The authors wish to express their gratitude to S. Prati, C. Gentili, S. Sacchetti, M. Maliosz, R. Vida, B. Pehrson, P. Szegedi, E. Daoukou, G. Heijnen, H. Benz, N. Whillans, A. Gavler, L. Andersson and R. Elverljung for valuable discussions.

REFERENCES

- [1] Hogg, R., Meckel, M., Stanoevska-Slabeva, K., Martignoni, R., "Overview of business models for Web 2.0 communities", Proceedings of GeNeMe 2006, pp. 23-37, Dresden, 2006
- [2] O'Really, T., "What is Web 2.0", <http://www.oreillynet.com/lpt/a/6228>, 2008
- [3] Foster, I., Kishimoto, H., Savva, A., Berry, D., Djaoui, A., Grimshaw, A., Horn, B., Maciel, F., Siebenlist, F., Subramaniam, R., Treadwell, J., Von Reich, J., "The Open Grid Services Architecture, Version 1.5.. Informational Document, Global Grid Forum (GGF), July 24, 2006, located at: <http://www.ogf.org/documents/GFD.80.pdf>
- [4] MacKenzie, M., Laskey, K., McCable, F., Brown, P. F., Metz, R., "OASIS – Reference Model for Service Oriented Architecture 1.0", <http://www.oasis-open.org/>, 2006
- [5] Schroth, C., "Web 2.0 versus SOA: Converging Concepts Enabling Seamless Cross-Organizational Collaboration", Proceedings of 9th IEEE International Conference on E-Commerce Technology and 4th IEEE International Conference on Enterprise Computing, E-commerce and E-Services, 2007
- [6] Fielding, R. T., "Architectural Styles and the Design of Network-Based Software Architectures", doctoral dissertation, Information and Computer Science Department, University of California, Irvine, 2000
- [7] ECMA-348, "Web Services Description Language (WSDL) for CSTA Phase III", 2nd ed., June 2004, <http://www.ecma-international.org/publications/standards/Ecma-348.htm>
- [8] SOAP/WSDL, WSDL, <http://www.w3.org/2002/ws/>
- [9] Figueiredo, R., Dinda, P., and Fortes, J., "A Case for Grid Computing on Virtual Machines", Proc. 23rd International Conference on Distributed Computing Systems (ICDCS), 2003
- [10] Barrato, R., Potter, S., Su, G., and Nieh, J., "MobiDesk: Mobile Virtual Desk Computing", MobiCom '04, Sept. 26-Oct., 2004
- [11] Barrato, R., Nieh, J., Kim, L., "THINC: A Remote Display Architecture for Thin-Client Computing", Technical Report CUCS-027-04, Department of Computer Science, Columbia University, July 2004
- [12] Talia, D., Trunfio, P., "Peer-to-Peer Protocols and Grid Services for Resource Discovery on Grids". In: L. Grandinetti (Ed.), Grid Computing: The New Frontier of High Performance Computing, Advances in Parallel Computing, vol. 14, Elsevier Science, 2005
- [13] Napster, <http://free.napster.com/>
- [14] Klingberg, T., Manfredi, R., "The Gnutella Protocol Specification," http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html
- [15] The Kazaa Network, <http://www.kazaa.com/>
- [16] Zhang, X., Liu, J., Li, B., Yum, T.-S. P., "DONet/CoolStreaming: A Data-driven Overlay Network for Live Media Streaming," IEEE INFOCOM 05, Miami, FL, Mar.2005
- [17] Zhang, M., Zhao, L., Tang, Y., Luo, J., Yang, S., "Large-Scale Live Media Streaming over Peer-to-Peer Networks through Global Internet", in Proceedings of ACM Multimedia 2005, Singapore, Singapore, 2005
- [18] The BitTorrent Project, <http://bittorrent.com>
- [19] Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S., "A Scalable Content-Addressable Network," ACM SIGCOMM 2001, San Diego, CA, Aug.2001
- [20] Khanvilkar, S., Khokhar, A., "Virtual Private Networks: An Overview with Performance Evaluation", IEEE Communication magazine, vol. 42 (10), pp. 146-154, October 2004
- [21] GENI—Global Environment for Network Innovations, <http://www.geni.net>
- [22] UCLP project, <http://www.uclpv2.ca/>
- [23] Bhatia K., ed., "Peer-to-Peer Requirements on The Open Grid Services Architecture Framework", Informational Document, Global Grid Forum (GGF), OGSAP2P Research Group, July 12, 2005, located at: <http://www.ogf.org/documents/GFD.49.pdf>
- [24] Li, J., Vuong, S., "A Scalable Semantic Routing Architecture for Grid Resource Discovery". Proc. 11th Int. Conf. on Parallel and Distributed Systems (ICPADS'05), vol. 1, pp. 29-35, 2005
- [25] Foster, I., "Globus Toolkit Version 4: Software for Service-Oriented Systems", IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, 2006, pp 2-13
- [26] Feamster, N., Balakrishnan, H., Rexford, J., Shaikh, A., van der Merwe, K., "The Case for Separating Routing from Routers," in ACM SIGCOMM Workshop on Future Directions in Network Architecture (FDNA), (Portland, OR), September 2004
- [27] Greenberg, A., Hjalmtysson, G., Maltz, D. A., Myers, A., Rexford, J., Xie, G., Yan, H., Zhan, J., Zhang, H., "A Clean Slate 4D Approach to Network Control and Management," SIGCOMM Comput. Commun. Rev., vol. 35, no. 5, pp. 41–54, 2005