Cloud Computing: The Next Revolution in IT

FRANK LEYMANN, Stuttgart

ABSTRACT

Cloud computing provides a new mode of use and of offer of IT resources. Such resources can be used "on demand" by anybody who has access to the internet. The resources are offered in a "utility-like" manner by providers based on actual-use-based prices. It is expected that cloud computing will change the way how organizations will use IT and think about IT: ultimately, cloud computing may relieve organizations from owning their own IT environment.
In this article we show that cloud computing can be seen as the next step in an evolution from isolated computers over clusters and beyond grids. We suggest a definition of clouds by abstracting their most important characteristics. The current set of cloud offerings is organized in a layered structure, and we propose an additional layer on top of that allowing to build applications in a composite manner ("composite as a service"). The distributed architecture of such composite cloud applications is derived by considering the structure of the individual services being composed. Finally, we argue that predefined points-of-variability are of utmost importance for cloud applications to be able to easily adapt them to the different requirements of the huge number of cloud customers.

1. CLOUD CHARACTERISTICS

The term *cloud computing* has been established recently to capture a particular use of IT resources (both, hardware and software). A couple of definitions of this term have been proposed (e.g. [3], [5], [16], [25]) but none of them has been generally accepted yet. In this article we suggest to define cloud computing by means of the following characterizing properties in using resources:

(i) Details of the resources (e.g. location, device type, software version) are not known (and often even not even of interest) to its users,
(ii) shortage of resources is no longer a concern for users (i.e. resources are available whenever a user wants to access them),
(iii) payments are due for the actual use of the resources (in contrast to periodic, constant rental fees).

These properties correspond to technologies or concepts which have been discussed individually since quite some time: (i) virtualization, (ii) elasticity, and (iii) utility computing. *Virtualization* basically means that an additional component within the software stack isolates the user of a resource from its concrete idiosyncrasies. Properties of individual resources are abstracted and collections of resources with the same type of abstracted properties are offered as a single large resource of that type.

Elasticity means that a resource is always available to the user, and that the resource grows or shrinks when more of the resource or less of the resource is needed by the user, respectively. Especially, elasticity implies that the environment providing a resource is high-available and scalable. An environment is called *high available* if it accepts requests and produces correct responses on a 7 days a week, 24 hours a day (7%24) basis; in practice, the 7%24 property (i.e. 100% availability) cannot be met and an environment is considered high available if it fails at most a few minutes per year (99.999% availability). *Scalability* of an environment refers to the property that the environment increases or decreases the resources used by the user on behalf of the user based on the user's actual demand for resources.

The property of elasticity implies that the actual amount of resources used by a particular user may change (significantly) over time. Especially, a user requests resources on demand, i.e. whenever a resource is actually needed and without any long-term indication about the future use of the resource: it is assumed that enough of it is "just there" whenever needed (*on demand computing*). In such an elastic environment users will only pay for the actual resources used ("pay as you go"). This usage of computing resources is similar to the use of power, gas, telephony etc. provided by public utilities; because of this, providing computing resources in such a manner is referred to as *utility computing* [28]. Utility computing and on demand computing are two sides of the same coin: utility computing represents the point of view of the provider of resources in the cloud, and on demand computing represents the view of users of resources in the cloud.

The primary reason for adapting cloud computing from a user perspective is to move from the model of capital expenditure (CAPEX) to operational expenditure (OPEX): Instead of buying IT resources like machines, storage devices, or software etc and employing personnel for operating, maintaining etc these resources, a company pays another company (the "provider") for the actual resources used (*pay-as-you-go*). An important aspect of this is that a company no longer needs to overprovision its IT resources: it is typical today, when a company invests in its own resources, that the amount of resources invested in corresponds to the maximum amount of resources needed at peak times – with the result that much of these resources are not needed at all during regular periods. Using cloud computing allows to use the resources actually needed without taking care about peak loads: this is because of elasticity property of clouds.

Providers of IT resources in the cloud exploit advancements in virtualization and provisioning technologies allowing to benefit from economies of scale: virtualization technology enables to share the same physical resources between different customers to significantly increase the utilization of each individual resource, thus, reducing the overall cost of use of a resource. Provisioning technology allows to automatically install, configure, deploy etc (and finally release) even complex application environments [21] also contributing to an optimized utilization of resources shared between the customers of the provider. Manual labor is no longer needed for creating and releasing such environments, further reducing the cost of providing IT resources. The corresponding savings make offering resources in clouds attractive for providers.

In addition, clouds are seen to be an excellent vehicle to sell even niche products (i.e. products that have only few potential customers in a given geographic region): when the cost of selling (offering, marketing, distributing, etc) of a niche product exceeds the profit gained, such a product is not economically viable. But especially software can be easily sold by means of a cloud: producers of software offer their products to cloud providers who in turn offer these products to their customers who may even run the software at the provider side (instead of installing, running, managing it on their own premise). As a consequence, the cost of selling software may be significantly reduced and niche software products may become a viable business [2].

Besides the use of cloud computing in commercial companies (e.g. [1]), exploiting the cloud computing paradigm in scientific applications has been investigate too [11]. Moving e-science applications to the cloud requires similar considerations about the advantages and disadvantages to moving business application areas to the cloud. Cloud computing will likely have a similar impact on e-science as it will have on corporate IT.

2. VIRTUALIZATION TECHNOLOGIES

Cloud computing may be seen as the next step in an evolution from isolated computers over clusters and beyond grids.

Virtualization technology has been used on single computers since decades [8]: the resources of one physical computer can be partitioned into logical resources and rearranged into so-called *virtual machines*, i.e. complete computers the ingredient resources of which are dedicated logical fragments of resources of one or more physical machines. An application together with its required middleware stack and even the required operation system can be hosted by a virtual machine. This results in a significant increase in utilization even of a single physical computer by allowing to run very heterogeneous application stacks on one and the same machine. Since applications run in separate virtual machines they are isolated from each other, thus contributing to the overall availability of the physical machine in case of crashes of individual applications. Furthermore, virtual machines can be cloned and moved from one physical computer to another, thus contributing to the overall scalability of the resulting environment. This requires that the format of the virtual machine images are the same on the source computer and the target computer; until recently, these formats have been proprietary but new standardization efforts support exchange of virtual machines even across different vendors ([12], [29]).

Clusters have been invented as a different means to increase (even further) the utilization, scalability and availability of individual computers [27]. A *cluster* is an interconnected set of complete computers (i.e. ones that could be used as standalone computers) that are perceived from the outside as one big computer. Since requests are made to the cluster, the concrete machine that will work on the request is not known to the requestor, i.e. the set of computers bundled into a cluster is virtualized. Consequently, a cluster ensures that the encompassed computers are evenly utilized. By adding machines to a cluster arbitrary scalability can be achieved (in theory). Since the computers that make up a cluster are indistinguishable the corresponding redundancy (plus recovery technology) ensures (high and continuous) availability. Together, a cluster provides virtualization and elasticity – although at different orders of magnitudes than what cloud computing provides. The focus of a cluster is to ensure an optimal use of the IT environment of an individual company in order to support the company's application mix.

Grids have their origin in the need for huge compute power required for solving large computational problems (especially in science) [13]. Nevertheless, the technology evolved to also support traditional, business applications [14]. A *grid* – like a cluster – is an interconnected set of complete computers perceived from the outside as a single computer. When compared with a cluster, a grid is (i) focused to support individual applications ("grand challenges") instead of a whole application mix of a given company, and (ii) is typically not owned by a single company but consists of computers own by organizations willingly to share their computers with others ("virtual organization"). This shows the origin of grids in scientific computing: researchers share their computers with other researchers in order to be able to use numbers of computers that an individual researcher can typically not afford; (parts of) this pool of computers can then be reserved by and assigned to a particular researcher for a period of time to solve a computational problem. Obviously, by assigning parts of the grid to a business application, a grid can also be used to support traditional information processing problems. Like clusters, grids offer virtualization and elasticity capabilities. But both technologies do not focus on the pay-as-you-go and utility aspects.

The virtualization of individual software functions has been addressed by *Web service* technology [29]. A Web service encapsulates an individual function and makes it available on the network

hiding all of the idiosyncrasies of its hosting environment. I.e. a user of a Web service is not aware of any implementation detail (programming language, operating system, application server, actual location, actual format of the data exchanged, and so on) of the function used. Thus, Web service technology provides virtualization of software functions: a user specifies only the kind of function needed and the data to be processed, and the Web service middleware (so-called *Enterprise Service Bus* [6]) finds an appropriate implementation of the function in the network, transforms the data according to the expected format, and communicates with the remote location and so on – without the user being aware of this happening [17]. Since the use of resources in a grid can be represented as software functions making these resources available which in turn can be rendered as Web services, Web service technology has been used to implement grids ([9], [10]). Thus, Web service technology is virtualizing both, software as well as hardware.

Supporting the virtualization of software and hardware, Web services provide the technological basis for utility computing and on demand computing [18]: users simply describe their required overall environments and the corresponding resources are made available by using the Web services that represent these resources. For that purpose, the descriptions of the required environment are translated into a series of Web service interactions that allocate the hardware needed, and then install, deploy and configure the corresponding software in the allocated hardware environment – jointly referred to as *provisioning* of the environment ([17], [18], [21]).

When comparing cloud computing with grid computing, the focus of a *cloud* is on providing arbitrary subsets of resources instead of only being allowed to request complete application environments. Furthermore, clouds make accessing "standard" types of resources (like storage, virtual images, particular application functions) as simple as possible at the price of losing the high degree of flexibility of allocating resources offered by grids. Clouds can be realized on top of grids, clusters, or directly on machines (perhaps using virtualization technologies on those machines) – see Fig. 1.

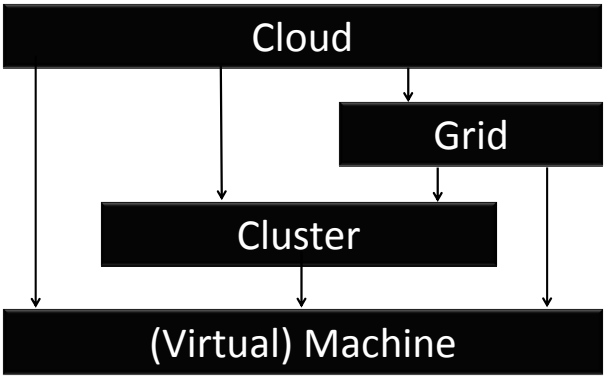

Fig. 1: Implementation Options for Clouds.

3. CLOUD SPECTRUM AND LAYERING

The spectrum of resources offered today in the cloud is vast and seems unstructured. At one end of the spectrum low level hardware like CPUs or network access is offered in the cloud. Next, storage of various types is made available, spanning from raw disks, over files and queues to databases. Complete computers can be accessed on the form of virtual machines in the cloud. The former kind of cloud services is often collectively referred to as *infrastructure as a service* (IaaS). Middleware to host applications in the cloud is offered too, called *platform as a service* (PaaS). Complete standard application packages can be used as services in the cloud, called *software as a service*

(SaaS). Finally, services in the cloud can be composed into new services which in turn can be provided in the cloud; we suggest to use the term *composite as a service* (CaaS) for that kind of cloud services. Note, that the different environments offered as a service are collectively referred to a *aaS ("anything as a service").

Many companies are familiar today with virtualization technology and cluster technology, some companies use grid technology today. The use of these technologies within companies is for optimizing the use of their on-premise resources. Thus, it seems likely that companies will start using cloud technology for the same purpose still on their own premise, i.e. without using cloud offerings from the internet (off-premise), without sharing the resources of the cloud with others. This kind of clouds is referred to as *private clouds* ([3], [25]). In contrast to this, clouds that share their resources with others are called *public clouds* ([3] , [25]). When being familiar with cloud technology in its private cloud variant, companies will start moving selectively to public clouds. This will allow to access additional resources if the resources of the private cloud do not suffice for peak loads, or if new kind of resources like new application functions appear on the public cloud that are not available on the private cloud, for example. The resulting topology of clouds is referred to as *hybrid cloud* (see Fig. 2 for the relations of these types of clouds). Hybrid clouds require standardization that supports interoperability between clouds of different providers [26]: Web service technology will play a major role in this. Based on the virtualization features of Web service technology, resources can be used in a transparent manner, i.e. the fact whether a resource is in the private cloud or in the public cloud can be hidden from the requestor.

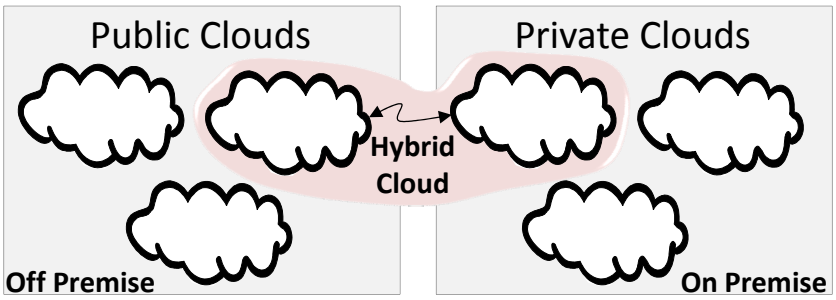

Fig. 2: Types of Clouds.

As sketch above, today's offerings of resources in the cloud are quite broad and diverse. In order to help understanding the potentials of clouds and to foster architectural thinking about clouds and their use, bringing structure into the spectrum of cloud offerings is needed. For this purpose, different bundles of services of clouds and corresponding layers have been proposed (e.g. [3], [16], [25]). We adopt the bundling of resources into the layers Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) (e.g. [25]), but we suggest to add a layer on top of this stack called Composite as a Service (CaaS) (see Fig. 3).

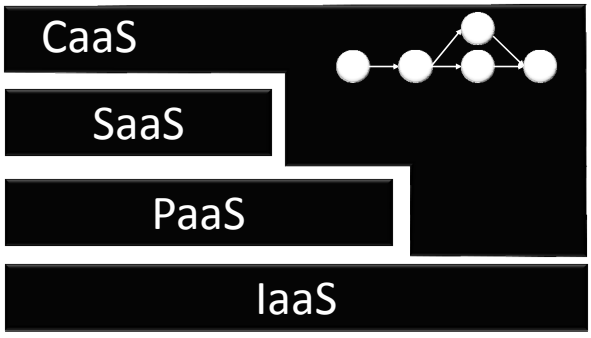

Fig. 3: Cloud Layers.

The CaaS layer applies the principle of process-based applications [19] followed today by most standard application software. This principle is also accepted in the domain of service oriented architecture (SOA) and is there referred to as *service composition* [29]. It is very likely that service oriented architectures will be the basis for clouds and their use, thus, service based applications will be the paradigm of future cloud applications. Since service based applications are typically service compositions, the notion of *composite applications* will be key for building applications in the cloud. A composite application makes use of services provided by various providers in different clouds and composes them into a new service. The established means to compose applications from existing services is via orchestrations [29], which basically means that the new service is described as business process the activities of which are realized by the other services. Since the orchestration is again a service the result is a recursive aggregation model for services in the cloud. In future, upcoming mechanisms like choreographies and service networks [4] will likely play a key role also in composing applications.

4. CLOUD APPLICATIONS

Applications typically consist of three layers: a presentation layer, a logic layer, and a resource layer [15]. Basically, the presentation layer is in charge of communicating with the requestor of the services offered by the application, the logic layer implements the domain logic provided by these various services, and the resource layer manages the resources (i.e. data etc) accessed by the logic layer. In practice, the different layers of an application run in different environments. Thus, it is only natural to make sure that an application, when running in the cloud, can be split the same way, i.e. having different of its layers running in different clouds making use of different *aaS environments (see Fig. 4). The proper positioning of the layers of an application within different clouds may even change dynamically during runtime (*application mobility*). For example, the logic layer of an application may be moved from the private cloud of the hosting company into the public cloud to serve unforeseen peak loads, and back from the public cloud to the private cloud when the load of the application becomes normal again. Consequently, hybrid clouds and the corresponding required interoperability of clouds will be fundamental to support this kind of scenario.

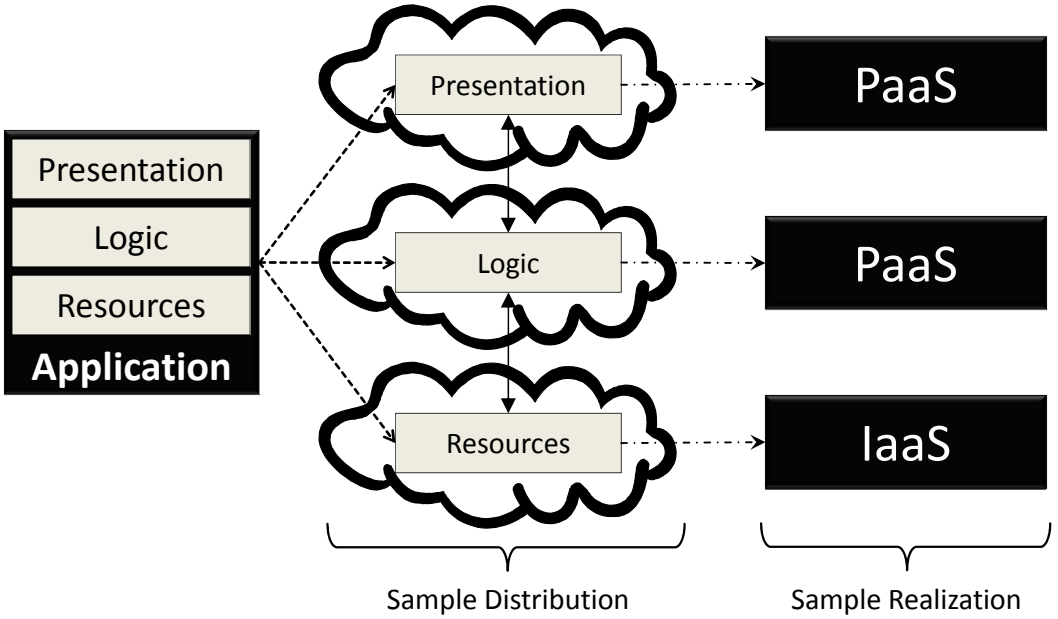

Fig. 4: Moving an Application to Clouds.

A composite application uses individual functions offered as services by other applications as implementations of the activities of the business process specifying the composition. This business process will be hosted in a CaaS environment in a cloud. The services used by it are implemented by applications hosted in other clouds and the layers of these applications may be distributed across different clouds hosted in different environments (see Fig. 5). Thus, the overall resulting configuration underlying a composite application may be very complex – which obviously result in many problems to be solved in the area of service levels, compliance, monitoring etc.

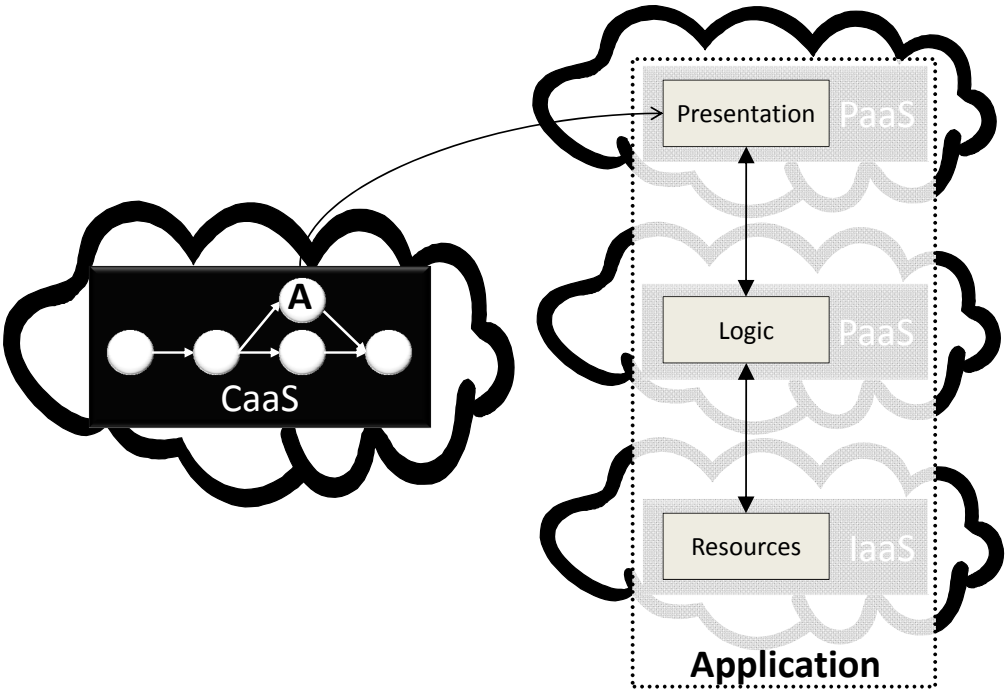

Fig. 5: Distributed Architecture of Cloud Applications.

Applications in the cloud will be able to be used by many different customers. As already observed today with standard applications, these customers will typically have diverse requirements on the

application in terms of non-functional properties (quality of services and service levels) as well as functional properties (e.g. to reflect a customer's internal business processes). Thus, cloud applications have to be *customizable*, i.e. they have to enable to be adapted to the customers' requirements. Since no non-trivial application can support arbitrary requirements, the kind of adaptability supported by an application has to be defined upfront by the creator of the application; this can be done by specifying corresponding *points of variability* (PoV) and their mutual dependencies [23].

To increase the number of possible users, applications have to support many different points of variability. Customizing such an application correctly considering all dependencies between the points of variability may become very complex. In order to support a user in customizing an application without violating such dependencies, processes can be generated out of the points of variability and their dependencies that guide a user through the customization steps [22]. Once the customization is done the customized application should be automatically provisioned. [21] suggests techniques that help towards that vision. Provisioning technology may even be used to make services available at runtime when they are needed [20].

One aspect of customization – called "multi-tenancy" – deserves special attention [7]. *Multi-tenancy* (in its true sense) refers to the fact that an application used by different customers must ensure that the data produced by one customer is complete isolated from the data produced by another customer. When the application runs on premise of each customer this is trivially satisfied. But when the application is hosted by a utility provider different options exist to realize multi-tenancy. A straight-forward realization provides completely separate environments (i.e. separate hardware and corresponding software stack) for each customer – now on-premise of the utility provider. But this mode of operation counters to the utility paradigm that assumes that resources (i.e. hardware and software stack) are shared between customers to mutually benefit from economies of scale. Some techniques of how to ensure multi-tenancy are available (e.g. [7], [24]).

5. CONCLUSION AND OUTLOOK

We have shown that cloud computing evolved from clusters and grids. Its defining characteristics are virtualization, elasticity, and pay-as-you-go pricing. Clouds make use of virtualization technologies like virtual machines and Web service technology. Clouds themselves will be exploited as private clouds and public clouds, as well as hybrid clouds as a combination of both. The services offered by clouds can be grouped into four major layers: infrastructure as a service, platform as a service, software as a service, and composites as a service. Applications today are typically structured in three layers themselves (different from the ones mentioned before: presentation layer, logic layer, and resource layer), which implies that these layers of an application should be able to run in different clouds hosted by different *aaS layers – many research problems result from that. This further complicates the structure of composite applications in the cloud, opening up another thread of research questions – monitoring, compliance service levels, component mobility to name but a few. The huge number of diverse customers of a cloud application put a high demand with respect to customizability on it. Again this implies difficult research problems like proper definition of the variability of an application, multi-tenancy of applications and corresponding middleware, automatic provisioning etc. Despite these many open problems, the benefits of cloud computing will make sure that this technology will get widely adopted with deep impact on the IT of organizations: IT is about to become the next utility – the future will be "cloudy".

Acknowledgement: I am very grateful to all my colleagues at the Institute of Architecture of Application System (IAAS) for the fruitful and stimulating discussions on the many aspects of building modern application systems. With respect to the subject of this paper, my special thanks goes to Ralph Mietzner who committed to spend a couple of years to the subject of cloud computing, thus, influencing my thinking on this subject by the many discussions we had.

6. REFERENCES

(All links checked on 3.7.2009)

[1] Amazon Web Services Case Studies, http://aws.amazon.com/solutions/case-studies/

[2] Ch. Anderson: "The Long Tail", Wired Magazin 12.10 2004, http://www.wired.com/wired/archive/12.10/tail.html

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia: "Above the Clouds: A Berkeley View of Cloud Computing" http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf

[4] M. Bitsaki, O. Danylevych, W. J. van den Heuvel, G. Koutras, F. Leymann, M. Mancioppi, Ch. Nikolaou, M. Papazoglou: "An Architecture for Managing the Lifecycle of Business Goals for Partners in a Service Network" Proc. ServiceWave 2008.

[5] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, I. Brandic: "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility", Future Generation Computer Systems 25 (2009).

[6] D. A. Chappell: "Enterprise Service Bus", O'Reilley 2004.

[7] F. Chong, G. Carraro: "Architecture Strategies for Catching the Long Tail", Microsoft 2006, http://msdn.microsoft.com/en-us/library/aa479069.aspx#docume_topic5

[8] R. J. Creasy: "The origin of the VM/370 time-sharing system", IBM Systems Journal 25(5), 1981.

[9] K. Czajkowski, D. Ferguson, I. Foster, J. Frey, S. Graham, T. Maguire, D. Snelling, S. Tuecke: "From Open Grid Services Infrastructure to WS-Resource Framework: Refactoring & Evolution", Fujitsu, Globus Alliance & IBM, 2004.

[10] K. Czajkowski, D. Ferguson, I. Foster, J. Frey, F. Leymann, M. Nally, T. Storey, S. Tuecke, S. Weerawaranna: "Modeling stateful resources with Web services", Globus Alliance & IBM, 2004.

[11] E. Deelman, G. Singh, M. Livny, B. Berriman, J. Good: "The Cost of Doing Science on the Cloud: The Montage Example" Proc. Super Computing 2008.

[12] DMFT Standard: "Open Virtualization Format Specification" Document Number: DSP0243, http://www.dmtf.org/standards/published_documents/DSP0243_1.0.0.pdf

[13] I. Foster, C. Kesselmann: "The Grid", Morgan Kaufmann 1999.

[14] I. Foster, C. Kesselmann: ”The Grid 2”, Morgan Kaufmann 2004.

[15] M. Fowler: ”Patterns of Enterprise Application Architecture“, Addison-Wesley 2003.

[16] S. Jha, A. Merzky, G. Fox: ”Using Clouds to Provide Grids Higher-Levels of Abstraction and Explicit Support for Usage Modes”, Concurrency and Computation: Practice and Experience – Special Issue of the OGF, 2009.

[17] F. Leymann: “The Influence of Web Services on Software: Potentials and Tasks”, Proc. 34th Annual Meeting of the German Computer Society 2004.

[18] F. Leymann: “Combining Web Services and the Grid: Towards Adaptive Enterprise Applications”, Proc. CAiSE/ASMEA 2005.

[19] F. Leyman, D. Roller: “Production Workflow”, Prentice Hall 2000.

[20] R. Mietzner, D. Karastoyanova, F. Leymann: “Business Grid: Combining Web Services and the Grid”, Transactions on Petri Nets and Other Models of Concurrency 2009.

[21] R. Mietzner, F. Leymann: “Towards Provisioning the Cloud: On the Usage of Multi-Granularity Flows and Services to Realize a Unified Provisioning Infrastructure for SaaS Applications”, Proc. International Congress on Services SERVICES 2008.

[22] R. Mietzner, F. Leymann: “Generation of BPEL Customization Processes for SaaS Applications from Variability Descriptors” Proc. International Conference on Services Computing, Industry Track, SCC 2008.

[23] R. Mietzner, F. Leymann, M. P. Papazoglou: “Defining Composite Configurable SaaS Application Packages Using SCA, Variability Descriptors and SaaS Multi-Tenancy Patterns”, Proc. 3rd Intl. Conf. on Internet and Web Applications and Services, ICIW 2008.

[24] R. Mietzner, T. Unger, R. Titze, F. Leymann: “Combining Different Multi-Tenancy Patterns in Service-Oriented Applications”, Proc. 13th IEEE Enterprise Distributed Object Conference, EDOC 2009.

[25] National Institute of Standards and Technology: “Draft NIST Working Definition of Cloud Computing”, 2009, http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v12.doc

[26] Open Cloud Manifesto, http://www.opencloudmanifesto.org/Open%20Cloud%20Manifesto.pdf

[27] G. Pfister: “In Search of Clusters”, Prentice Hall 1998.

[28] M. A. Rappa: “The utility business model and the future of computing services”, IBM Systems Journal 43(1) (2004).

[29] S. Weerawarana, F. Curbera, F. Leymann, T. Storey and D. F. Ferguson: “Web Services Platform Architecture”, Prentice Hall 2005.

[30] VMware, XenSource: “The Open Virtual Machine Format Whitepaper for OVF Specification”, 2007, http://www.vmware.com/pdf/ovf_whitepaper_specification.pdf