Aspects of Data-Intensive Cloud Computing

Sebastian Frischbier and Ilia Petrov

Databases and Distributed Systems Group
Technische Universität Darmstadt, Germany
{frischbier,petrov}@dvs.tu-darmstadt.de

Abstract. The concept of Cloud Computing is by now at the peak of public attention and adoption. Driven by several economic and technological enablers, Cloud Computing is going to change the way we have to design, maintain and optimise large-scale data-intensive software systems in the future. Moving large-scale, data-intensive systems into the Cloud may not always be possible, but would solve many of today's typical problems. In this paper we focus on the opportunities and restrictions of current Cloud solutions regarding the data model of such software systems. We identify the technological issues coming along with this new paradigm and discuss the requirements to be met by Cloud solutions in order to provide a meaningful alternative to on-premise configurations.

1 Introduction

Large-scale, internet-based and data-intensive services have recently become a cornerstone of our daily activities: we buy books on Amazon, sell things on Ebay, stay in contact with friends and colleagues via Facebook, Googlemail or Hotmail, work collaboratively on documents with ThinkFree or GoogleDocs, comment on videos via YouTube, share our snapshots on Flickr or Picasa and plan our journeys with GoogleMaps [1,2,3,4,5,6,7,8,9]. These are just examples to name a few well-known internet-based and data-intensive services we use in our everyday life. All these services are powered by highly scalable applications based upon massive computing infrastructures. Until lately, operating infrastructures to power online-services at this scale involved large up-front investment and running costs for any provider: Whole data centers (or at least sufficient resources within) had to be located, acquired, staffed and operated. The whole technology stack had to be set up and optimised. Even if these resources are not run on-premise but rather as a managed-hosting solution the contracts are usually fixed and not usage-based, shifting the risk of resource planning from the infrastructure provider to the service provider. There has always been the danger of ill-sizing one's infrastructure based on wrong assumptions about the demand to be. Today's Internet is the backbone of global communication, connecting millions of users 24 hours seven days a week. This results in complex patterns of demand that are hard to anticipate and influence. On the one hand this offers the opportunity to attract millions of customers all over the world within hours, sometimes even by a minimum of controlled advertisement. On the other hand, this could backfire as well: unavailable, slow or insecure services get a bad reputation quickly, leading to a drop in demand and a loss of revenue [10]. In recent years there has been an increase in different commercial offerings, all being subsumed under

K. Sachs, I. Petrov, and P. Guerrero (Eds.): Buchmann Festschrift, LNCS 6462, pp. 57–77, 2010.

the keyword Cloud Computing. Due to these solutions offered by Amazon, Google, Microsoft and others, highly-scalable standardized computing power on-demand seems to become a commodity. The promise is to enable nearly everyone to build and operate powerful applications easily without the risks mentioned above. No up-front investment is needed and the risk of ill-sizing is shifted back from the service provider to the infrastructure provider. Thus Cloud Computing seems to be the future architecture to support especially large-scale and data-intensive applications. If the reality of Cloud Computing lives up to its promises we believe this movement to trigger two trends reinforcing each other: on the side of service providers, the financial benefits of Cloud Computing will lead to an overall adoption by companies of all sizes to deliver highly scalable services with as minimal costs as possible. On the user side, the broad supply of online-services raises the bar of expectations regarding usability, availability and reliability of new products even more. Thus companies will have to invest even more into their infrastructure in order to meet their customers' raised expectations. This again may lead to relying even more on Cloud Computing. Alas, on-premise middleware with individual frameworks atop is the dominating architecture for those large-scale, data-intensive applications by now. Especially in the context of Enterprise Computing, extensive middleware solutions are still custom-tailored to the requirements of each application. In turn, the application's functionality and design rely heavily on the properties and requirements of the supporting infrastructure. Although middleware components such as databases, application servers or messaging systems are highly standardized components yet, their individual combination and configuration allows to support highly specialized data-models required by the application logic. To us, this close symbiosis of application and infrastructure is still an obstacle to the broad adoption of Cloud Computing by large enterprises. Switching from traditional middleware to the Cloud has a massive influence on the management of an application's life-cycle. Among the several concerns are matters of design, implementation, testing, deployment, version control, debugging and maintenance. In this paper we focus on the underlying technological issues affecting the architectural decisions at the initial phase of a data-intensive application's life-cycle. We argue that there are certain requirements of data-intensive applications that are hardly meet by today's Cloud Computing. As a result, one still has to build and manage certain infrastructures on-premise in order to meet all requirements. Hence we first introduce the concept of Cloud Computing with its main characteristics, enablers and properties of today's solutions in Sect. 2. The requirements of data-intensive applications are then exemplarily presented in Sect. 3. Afterwards we contrast the mentioned properties of today's Cloud offerings with the requirements regarding data-models of two types of data-intensive applications in Sect. 4. In Sect. 5 we present a selection of academic as well as non-academic research results and sum up our findings in Sect. 6.

2 What Cloud Computing Offers Today

Although being a buzzword widely used in marketing and academia, the definition of Cloud Computing is still somehow blurred. There have been many attempts to define it, leading to multiply defined keywords which may seem contradicting at first [11]. Thus we sum up what we think is the core of Cloud Computing in this section, deducing

the taxonomy being used in this paper to avoid any confusion. The section concludes with a discussion of several provider related as well as user related technological issues regarding the paradigm of Cloud Computing.

2.1 What Is Cloud Computing?

To cover the basic principles of Cloud Computing we refer to the definition of Cloud Computing by now being most agreed on. It was introduced by the National Institute of Standards and Technology (NIST) in 2009 [12]:

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

Hence, the term Cloud Computing does not refer to a single technology or specific product. It rather denotes a generic term covering all solutions that offer computing resources as a commodity. The analogies quite often used are those of power or water supply. Another quite fitting analogy approach was introduced by Thomas Bittman of Gartner in 2009 [13]: transmitted music. The first transmission of music over air started as a very basic service with poor and unreliable quality. Anyway it has been an interesting and quite cheap alternative to the phonographic cylinders used as storage back than. Over time, standardization set in, quality and supply increased. Although these analogies have their weak spots we think it is an interesting idea worth mentioning. Based on their broad definition, Mell and Grance deduce five abstract key characteristics of Cloud Computing all current industrial offerings have in common [12]:

- *On-demand Self Service* In contrast to the provision of resources in on-premise or managed hosting scenarios, no human interaction is necessary to adjust the quantity of resources needed in the Cloud. This is true for provision of additional resources (scale-out) as well as for the release of unused ones (scale-in).
- *Broad Network Access* The service requested is delivered over a standardized network, typically the Internet.
- *Resource Pooling* The resources needed to provide the requested service automatically to the user are drained from a resource pool designed to serve dynamically multiple users (multi-tenancy). Size, location and structure of this resource pool are concealed to the user as well as the identity of the different parties being served by the same physical resource.
- *Rapid Elasticity* The provision of resources from the pool as well as the release of unused resources back to the pool has to be applied rapidly at any quantity and time. This creates the illusion of infinite resources to the user.
- *Measured Service* The performance and usage of resources are monitored and metered automatically in order to optimise the overall usage as well as to provide the information necessary for usage-based billing.

In regard to the parties having access to and control over the resource pool, public, Private and Hybrid Clouds are to be distinguished: Public Clouds are accessible by the

general public but are controlled by a single dedicated Cloud provider. In this scenario, the tenants are independent organizations or users sharing the same resources. Private Clouds denote a scenario where Cloud services are only accessible for divisions of a single client's organization. From this point of view, a Private Cloud could be seen as a Public Cloud within a single company. The control over the resource pool is either on-premise or off-premise (managed-hosting solution). Hybrid Clouds combine resources of Private and Public Clouds, hence allowing the quick scale-out of a Private Cloud with resources temporarily drawn from a pool being publicly accessible.

2.2 Comparing Cloud Computing to Related Paradigms

Without first contrasting Cloud Computing to other related paradigms it is hard to point out and evaluate any unique contributions of this new paradigm. Thus we compare the key characteristics of Grid Computing, Utility Computing and Software-as-a-Service (SaaS) with those of Cloud Computing. The concept of Grid Computing emerged in the 1990s to provide an alternative for organisations to obtain computation power hitherto only delivered by supercomputers [14]. As Foster points out in [11] Grid Computing aims at delivering abstracted computational resources drawn from a distributed inter-organizational resource pool. Each participating organisation remains in control of the innate resources being committed to the pool while earning the right to use the Grid as a whole. A multilayered fabric is responsible for abstracting the underlying resources and scheduling the usage accordingly by queuing. Foster introduces a three point checklist to tell Grid and non-Grid infrastructures apart [15] by stating that a Grid (i) coordinates resources that are not subject to centralized control; (ii) is using standard, open, general-purpose protocols and interfaces; (iii) delivers nontrivial qualities of service. Applying these criteria as well as the characteristics mentioned above on Cloud Computing leads to the following conclusion: Grid computing and Cloud computing are not precluding but rather intersecting paradigms as depicted in Fig. 1. Both paradigms aim at delivering abstracted computing resources but differ in their particular approach and subject [11]: (i) Cloud computing aims at serving multiple users at the same time while Grid Computing is intended to deliver functionality at a scale and quality equivalent to supercomputers via a queuing system; (ii) Grids consist of resources owned and operated by different organisations while Clouds are under a single organisation's control; (iii) Cloud services can be obtained by using a standardized interface over a network while Grids require running the Grid fabric software locally. Both Utility Computing and Software-as-a-Service (SaaS) are rather generic terms. Mäkilä et al. define SaaS as a software deployment model, where the software is provisioned over the Internet as a service [16]. Utility Computing is even broader referred to by Ross and Westerman [17]:

> We define utility computing as a collection of technologies and business practices that enables computing to be delivered seamlessly and reliably across multiple computers. Moreover, computing capacity is available as needed and billed according to usage, much like water and electricity are today.

Taking all these top level definitions into account we propose the following categorization of the paradigms being discussed in this section: All concepts mentioned aim at

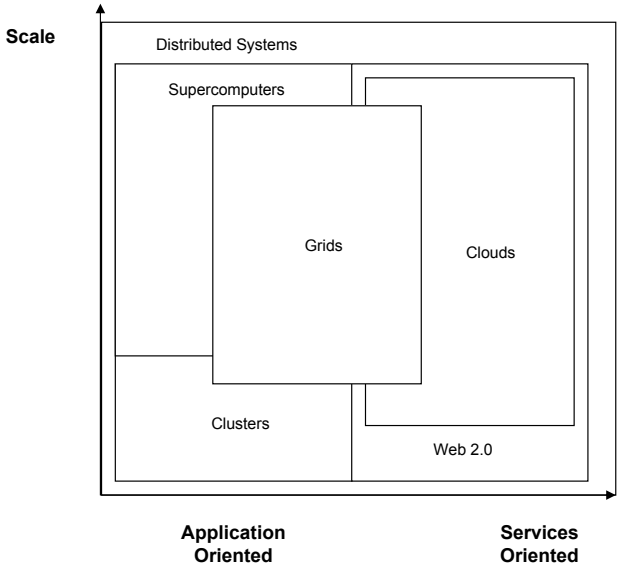

Fig. 1. Relationship of Grid and Cloud Computing as illustrated in [11]

delivering computational functionality as a service. While Utility Computing describes the general approach, the other concepts focus on different ways of realisation. SaaS focuses on delivering application logic while Grid Computing focuses on delivering computational power being equivalent to supercomputers. Thus Cloud Computing embraces these approaches, itself not being restricted to certain functionalities all the while being more palpable than Utility Computing.

2.3 Technological and Economic Enablers

To us, the observed characteristics of Cloud Computing as well as the success of this concept are the result of several economic and technological enablers reinforcing each other. We believe, that understanding these enablers and their connections is crucial to identify the real impact Cloud Computing has on large-scale, data-intensive applications. Thus we start by introducing the five technological and five economic enablers we think are the stepping stones for the success of Cloud Computing so far. Figure 2 illustrates the different enablers and their connection.

Technological Enablers

– *Virtualisation* is the key technology to enable Cloud Computing efficiently on the resource management level as well as on the product feature level. Regarding the resources being operated by the Cloud provider, this approach permits mainly three things: (i) the utilisation of a single physical device is maximised by hosting several virtual machines on it; (ii) the overall utilisation and power consummation of

the resource pool can be optimised by running as many instances as possible on as few physical devices as necessary to ensure the availability and performance guaranteed by strict Service Level Agreements (SLA) [11]. The aim is to turn machines automatically off if they are fully idle; (iii) maintenance cycles of hardware and disaster recovery actions are simplified due to the fact that VM instances can be shifted from one physical device to another or be restarted on uncorrupted hardware quickly. Regarding Cloud Computing offerings on the infrastructure level, virtualisation permits the product feature most referred to: Automated scaling. Vertical scaling denotes the resizing of a single VM instance given (e.g. adding and removing RAM or CPU respectively). Horizontal scaling refers to adding or removing additional instances of a given VM running in parallel. By using virtualisation, both types of scaling could be implemented efficiently. Based on an initial VM image, additional instances are launched automatically (horizontal scale-out). In case of vertical scaling, a dedicated VM instance is shut down to be automatically launched again with new parameters setting the desired system properties.

- *Grid Computing* is closely related to Cloud Computing in many aspects. Both are partially overlapping but are far from being identical as we have seen in Sect. 2.2. Nevertheless, the realisation of Grid Computing faces several technological challenges that hold true for Cloud Computing on the infrastructure level as well. Among them are the discovering, scheduling, brokering, assigning, monitoring and load balancing of resources as well as the core communication and authentication in distributed resources [11]. By addressing and overcoming these challenges for Grid Computing, technical solutions emerged that are now used to form the basic fabric on the infrastructure level of Clouds [14].
- *Service-oriented Architecture (SOA)* may be seen as the latest representative of design concepts for distributed systems leading to a paradigm shift in application design in this context. The widespread appliance of this paradigm has laid the general basis for heavily distributed applications in the Cloud [12]: Individual functions are encapsulated as autonomous services being accessible only through an implementation independent interface. The use of shared memory is substituted by message-based communication hence adding transparency of location to the transparency of implementation. New functions can be implemented by combining existing services using message-based interactions. By now there are already several solutions to challenges such as semantic interoperability, state-preservation or parallel computing that have to be solved wile pursuing this design approach. Only applications being based on these principles could benefit from Cloud Computing to the full extent.
- *Rich Internet Applications (RIA)*. For many years, the user experience of web based applications has been far from that of desktop applications. The ever increasing functionality of today's web technology (e.g. client-side processing, asynchronous communication or built-in multimedia support of web browsers) now permits complex web based applications that resemble desktop applications (c.f. ThinkFree or GoogleDocs). Using RIA to deliver the functionality of complex software over the Internet has certain advantages for both user and provider: no software has to be installed and maintained on the user's system because the application is hosted centrally by the provider and accessed by a web browser. Such centrally managed

software allows easier maintenance, resource provisioning and usage metering by the provider.
- *Broad network coverage*. To deliver high-volume products and smooth interactions for RIAs over the Internet, the overall coverage of broadband connections is crucial. This holds true not only for a good end user experience but for the interactions between different Cloud services and data centers as well. In recent years there has been a significant increase, not only in the overall coverage of broadband networks all over the world but also in capacity and performance.

Economic Enablers

- *Economy of Scale*. The characteristics of Cloud Computing we have seen so far favour large-scale infrastructures. As Greenberg at al. [18] or Church et al. [19] point out, especially providers of Cloud infrastructure services benefit from economy of scale at multiple cost categories: From one-time investments for land, buildings, power and network infrastructure to frequently returning costs for hardware and running costs for power supply, cooling and maintenance [20]. Especially the rapid decrease of hardware costs intensifies the economies of scale for infrastructures.
- *Multi-tenancy*. Running multiple users on a single resource is not a quite new concept from a technological point of view. Nevertheless, running different external customers in parallel on abstracted resources based on a shared and brokered infrastructure pool with isolated data, performance and application logic embraces certain challenges [21]. Besides the technological issues to ensure such isolation, multi-tenancy has to be accepted by customers.
- *Micropayment*. To exploit a sophisticated willingness to pay a provider has to offer different products in high granularity to different customers. Hence, charging even small quantities effectively with micropayment enables the provider to maximizing both revenue and utilisation. We subsume the emergence of different online payment methods (e.g. PayPal) as well as the increased use of credit card payments under this topic.
- *Open Source Software*. License fees for software have always been a serious matter of expense especially at large-scale data centers. Using open source software today can be an adequate way to decrease the impact of license fees on the revenue dramatically. By now, there are open source substitutes all along the technology stack, starting at the virtualisation layer and ending with the presentation layer (i.e. web browsers). Furthermore, open source software can be altered to suit very specific needs without violating license terms. Although that may be true for other proprietary systems as well, open source systems suit another long-term aspect: By relying on open source software (including open standards) lock-in effects to specific providers are minimised.
- *Standardisation*. Economy of scale and standardisation are closely related: Standardisation fuels economy of scale by making components interchangeable and comparable. In regard to Cloud Computing, standardisation helps to reduce complexity and increase flexibility on all levels of the technology stack (e.g. standardised hardware, application servers or frameworks).

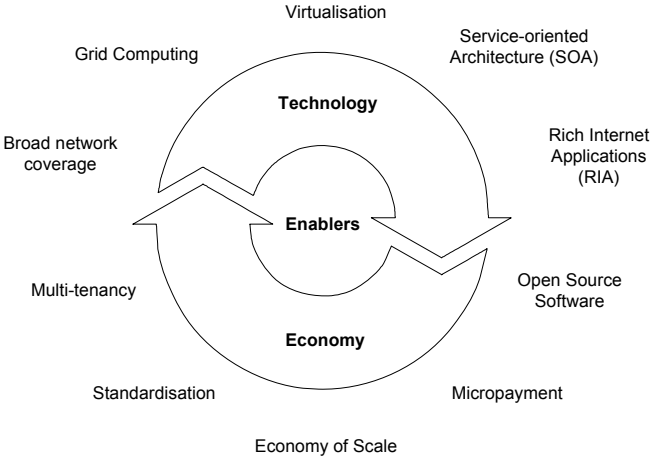

Fig. 2. Identified key enablers for Cloud Computing

Having introduced these enablers so far we point out some mutual dependencies between them leading to the characteristics of Cloud Computing as presented in Sect. 2.1: Virtualisation and Grid Computing form the fabric that enables resource pooling and rapid elasticity. SOA with its related paradigms reshaped both middleware and application architecture to run on those abstracted distributed resources efficiently as managed services. Economy of scale, standardisation and open source software make Cloud services cost-effective by rewarding scale, reducing complexity and minimising license fees as well as lock-in effects. RIA and broad network coverage simplify renting and maintaining computational functionality as on-demand self-services. Micropayment permits effectively charging the usage of such functionality at fine granularity, making even highly diversified products lucrative. Multi-tenancy maximizes both utilisation and risk-reduction by involving different parties. Users benefit from transparent pricing models based on metered usage and seemingly indefinite resources being provided to them on an abstraction level being suitable for their own purpose. Hence, further technological enhancements will be encouraged, fuelling the economic mechanisms mentioned here.

2.4 Resulting Business Models and Pricing

While presenting the characteristics as well as the enablers for Cloud Computing, we referred to the digital product being delivered only as abstract computing functionality. Substantiating this generic term leads to classifying the different delivery models of Cloud Computing and their business models. As we have discussed earlier, Clouds benefit highly from standardisation in order to maximize the economies of scale. Hence providers of Cloud services tend to standardise as many components of their products and supporting architecture as possible. Such standardisation is often achieved by adding an abstraction layer and restricting access to the technology stack beneath it. Therefore,

each delivery model can be defined by two aspects: A core functionality being offered as a service and a ratio of control by user vs. control by provider for the entire technology stack.

- *On-premise*. The whole technology stack is owned and controlled by the user. Until recently this is the typical setting for large-scale data-intensive applications as described in Sect. 1. This scenario could also be used to implement a Private Cloud setting with the technologies discussed in Sect. 2.3.
- *Managed Service Hosting (MSH)*. Entire servers or virtual machines are rented exclusively to a single user. The user can choose the operating system to be used and is in control of layers atop the infrastructure while the provider is responsible for maintenance and operation. The contract period is usually fixed and usage independently. Scale-out could be achieved by additional contracts and is seldom promptly.
- *Infrastructure as a Service (IaaS)*. As in MSH, the product core being delivered is computation power at the infrastructure level for the user to build his own stack upon. Again the user has full control over the layers atop of the infrastructure. In contrast to MSH, the resources being rented by the user are not dedicated servers but virtual machines or equally abstract machine instances based on a shared resource pool. The structure, location and usage of the resource pool are opaque to the user and organised according to the requirements of the provider. This has two advantages: Firstly, scale-in as well as scale-out of instances can be achieved in almost real time according to demand. Secondly, maintenance cycles can be performed by the provider without affecting the availability and performance guaranteed to the user. Pricing is usage-based, generally derived from the number, configuration and usage duration of the active instances.
- *Platform as a Service (PaaS)*. The product core is middleware functionality ranging from databank management systems (DBMS), messaging or load balancing to even more specialized functions to build custom applications upon. The functionality is provided through an abstracted application programming interface (API) making the implementation and detailed configuration of the underlying infrastructure inaccessible to the user. Thus, applications build on PaaS have to take that into account. When the usage for the application upon the platform grows, the provider is responsible for automatically scaling the underlying resources to deliver the quality and performance guaranteed. Pricing is usage-based with the reference value depending on the functionality being consumed.
- *Software as a Service (SaaS)*. The product delivered by SaaS is typically a fully-fledged application. Contrary to PaaS, services at this level mostly aim at end-users by providing high-level user-experience via rich internet applications. As SaaS is the first primarily end-user oriented Cloud business model introduced here, it is quite often identified with Cloud Computing itself by the public. All examples given in Sect. 1 refer to typical SaaS offerings. In addition to the graphical user interface provided by RIAs, advanced functionality can be accessed through APIs quite often to build custom extensions (e.g. as plug-ins for web browsers). Again the provider is responsible for scaling the underlying resources according to the utilisation of the application. Pricing is usage or subscription based (recurring fees).
- *X as a Service (XaaS)*. By restricting the user control even more towards the mere passive use of individual and highly specialised functionalities, new business

models can be derived. We propose here exemplarily the idea of Function as a Service (FaaS) as a special case of SaaS. Unlike SaaS, FaaS focuses on delivering a single functionality through a standardised API only. Examples would be the typical web services: Information about stock exchange prices, appraisal of creditworthiness, weather forecasts or visualisations of GPS coordinates. To us, the further specialisation of SaaS is motivated by the increasing occurrence of smartphone-based applications (apps) over the last two years. Many of these applications are specialised in visualising real-time information being delivered by large-scale online-applications. The throughput of today's mobile connections is still rather limited by way of comparison to broadband networks. In addition, many smartphones still have certain restrictions regarding screen resolution or supported technology. Hence the transmission of core information is often preferred to the use of extensive rich internet applications as in SaaS offerings. In such a scenario, the user would have even less control over the service consumed while the level of abstraction is extremely high. Driven by the trend of highly specialized apps for smartphones we believe the business model of XaaS/FaaS soon to grow out of the status of being a subtype of SaaS.

Although not being a prerequisite, it is often assumed that Cloud services are stacked to leverage the advantages of Cloud Computing at subjacent layers [22,14]. In addition, either IaaS, PaaS, SaaS or XaaS can be implemented as Public or Private Cloud. To us, the attractiveness of the business models mentioned above could be mapped to the Long Tail of users as depicted in Fig. 3. We base our assumption on the following reasoning: Moving from the first to the last business model both standardisation and specialisation

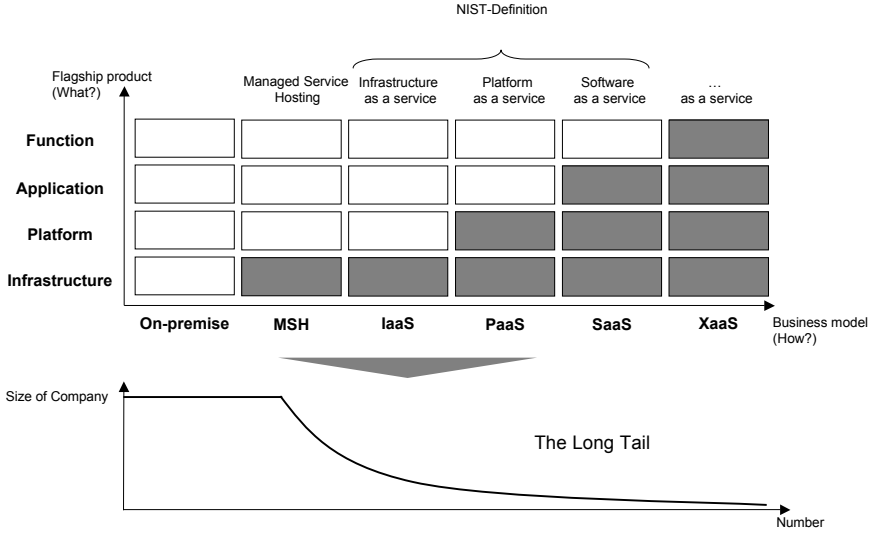

Fig. 3. Different business models of Cloud Computing derived from the level of abstraction and product core

increase. This results in offerings of higher granularity and cost effectiveness. Especially small and middle businesses (SMBs) or start ups may rank cost effectiveness and ease of use over extensive control.

2.5 Current Commercial Solutions for Public Clouds

To cover the range of Cloud solutions available today we present a selection of commercial solutions and their main characteristics in this section.

- *Amazon* offers solutions at all levels of abstraction. Nevertheless the dependencies of most offerings refer to a focus on the IaaS and PaaS levels. Thus we concentrate our brief presentation on representatives from both core and complementary solutions on these levels. With Elastic Cloud Computing (EC2), Amazon offers the archetyp of IaaS Cloud Computing by providing abstract computing facilities in terms of virtual Amazon Machine Images (AMI) [23]. Amazon Simple Storage Solution (S3) resembles a key-value storage solution for large objects that are seldom written but often read. The Relational Database Service allows less complex relational models. CloudFront allows to redirect requests to data stores geographically close to the customer. Other solutions at the IaaS level (i.e. Elastic Load Balancing,Virtual Private Cloud), the PaaS level (i.e. Elastic Block Storage, MapReduce) or as SaaS level (i.e. Cloud Watch,Flexible Payments Service) can be seen as extensions to these core products. For more detail on these solutions as well as use cases we refer to [24,25,26].
- *Google* aims at the layers atop of IaaS with several offerings: With Google App Engine [27], Google provides a PaaS solution to develop, test and host applications written in Java or Python. By developing applications against the Google App Engine API, issues regarding deployment, sandboxing, instantiation, scaling or load balancing are handled by the Google middleware and infrastructure. Google App Engine offers a schemaless datastore with an SQL-like syntax to manipulate tables while not being a traditional relational database. On the downside, there are artificial restrictions to the application's capabilities by the platform as mentioned in [14]. Other offerings such as Google Docs or Google Mail are most prominent rich internet applications with additional API access [28,29,30] as SaaS solutions. Google Docs offers the functionality of viewing and editing documents online by using a rich internet application while Google Mail implements a typical mail client. In addition, there are several experimental offerings that could be assigned to XaaS/FaaS: Google Finance Data API, Google Friend Connect APIs, Gadgets API, Google Geocoding API or the Google Prediction API. All matured offerings mentioned are available with basic functionalities being free of charge. Charging for the Google App Engine is usage-based and derived from the amount of storage and bandwidth as well as the CPU cycles used by the application. Google Docs and Google Mail are also offered subscription-based per user as professional versions.
- *Microsoft* entered the Cloud Computing market with Windows Azure in 2009. The Azure platform can be seen as a mixture of IaaS and PaaS by combining aspects of both [31]. Azure itself contains of offers several Cloud based storage solutions: Windows Azure Storage and Azure SQL. Windows Azure Storage is provided by

the Azure platform itself and consists of Blobs, Tables and Queues as native data models. A Blob stores binary objects up to 50 GB total thus being suitable especially for multimedia objects. Different Blobs are logically combined and managed within a Blob Container. Both Container and single Blob can be associated with metadata in form of key-value-pairs. Tables consist of entities (rows) and associated properties (colums) to support structured data. Queues represent a asynchronous messaging facility. Windows Azure SQL resembles a auto-scalable SQL-server as SaaS solution, claiming to work seamless with on-premise installations of Microsoft SQL Server [32].

- *Salesforce* is an on-demand CRM product that is traditionally regarded as one of the first in the field of Cloud Computing. In the meantime Salesfore supports wide range of applications and supports a robust Cloud platform, powerful application development framework and solid services. It a typical representative of the SaaS group. As underlying technology Salesforce uses Oracle RAC [33] as a relational database. The data model is simple and essentially comprising database tables and indices. Salesforce has a multi-tenant architecture and assigns a tenant to a Salesfore Node. The system has a powerful application development framework and and run-time environment. Applications are developed in Salesforece's own APEX programming language, which supports multi-tenant constructs natively, is dynamic and eases application and data construct generation at run-time.
- *Private Cloud Solutions* are offered by several suppliers. Among them are IBM, Sun Microsystems, Red Hat, Jboss, Oracle and Zimory.

2.6 Technological Issues to Think about

Some of the major issues that distinguish Cloud Computing from other approaches are: scale and simplicity. Under scale we mean: (i) the number of concurrent and potential users and requests per second; (ii) the size of the computing facility (CPUs, aggregate memory, aggregate storage capacity) and the size of the data; (iii) the geographical distribution. Simplicity, on the other hand, is associated with the way the system is used: (a) ease of programming API simplicity; (b) ease of administration and growth; (c) ease of consuming and combining different services. These are not easy to achieve especially under the premise of high performance. Hence most of the vendors and researches mingle Cloud applications (sometimes services) with Cloud infrastructure. Indeed the application requirements influence the way the Cloud infrastructure is designed and vice versa objective facts in the Cloud environment influence hard design choices in the Cloud application architecture. None the less the goal of many Cloud vendors is to provide a general platform on top of which Cloud applications can be developed. Below we discuss some of the issues specific to provider and application separately.

Provider Related

- *Consistency*. The responsibility of enforcing consistency of Cloud data is separated between the infrastructure and the application. In enterprise computing the established approach is expect declarative or implicit consistency such as in ACID transactions. Since at this stage most of the Cloud applications are tightly coupled with

their infrastructure the application developers implement application specific consistency mechanisms. Due to issues resulting from scale most of the consistency notions (two phase commit protocol, global atomicity) are inefficient or very difficult to realize. Due to the CAP theorem enforcing consistency (on infrastructural and application level) depends on handling availability and the required network resilience.

- *Availability*. Cloud services/application as well as the Cloud infrastructure are expected to be as available as enterprise applications. There are however several issues that need to be overcome: (i) internet reliability - all Cloud services are available over the internet hence the upper bound to their availability is the internet availability, which is max. 99.9%. The issue can be generalized in terms of network reliability. (ii) hardware fault tolerance since the Cloud utilises commodity hardware the infrastructure has to tolerate high failure rates. For example a recent analysis of the RAM failure rates at Google [34] showed that these lie higher than the currently assumed value. (iii) All these explain why whole data centers can be suddenly detached from the rest of the infrastructure. Methods to address these issues at present are: (a) load balancing; (b) redundancy in terms of (geographical) replication. Careful data placement minimize the response times; having multiple consistent replicas of the data helps to increase the availability (the request can be served on a different site). Careful data placement also helps the Cloud designers to fight latency. While the infrastructure can provide replication mechanisms these only work well if they are suitable to and well instrumented by the application. Latency is another issue that has to be addressed in Cloud applications. (its effect on enterprise applications is not as direct). It can be minimized through careful data placement so that requests can be served close to the user and through extensive caching. Both means to address latency are very application dependent.
- *Predictable Performance + Elasticity*. Since many resource hungry applications are sharing the same set of computing resources in a Cloud Computing facility, it should be guaranteed that there is no resource contention and that applications are isolated form each other and are provided with predicable framework performance. Two factors should be specially emphasized: elasticity and scalability. Elasticity is associated with providing stable and predicable performance as resource demands of applications increase (or decrease) ensuring optimal resource utilisation. If an application grows resources are not only added in terms of new servers, it also means more bandwidth, more storage, better load balancing. Elasticity is related to up- and down-scaling but is more complex in that it involves the Cloud infrastructure as well. Scalability is the relevant term in this context: that data store must be able to execute large requests with low response times and redistribute data an load on the new hardware. Some of the Cloud models (such as IaaS, i.e EC2) rely heavily on virtualization. While virtualization offers significant advantages in handling elasticity and resource sharing the way Disk IO and Network IO are handled. In fact [35,36] report similar results.
- *Multi-Tenancy*. Multi-tenancy is one of the key data storage concepts on the Cloud. It enables user data manageability, common storage and increased performance.

Every user is being seen as a tenant in a multi-tenant store: its data is stored in a common and general schema with other users' data. The data entries for a specific user are automatically annotated with the TenantID. Many Cloud solution derive tenant specific typed tables from the common data storage.

- *Scalable and High Performance Storage*. Due to many technical issues data storage is one of the central issues in Cloud Computing. In Cloud scenario it is one of the key factors for well-performing and scalable Cloud applications/services. While raw storage is needed in enterprise computing, it is an inadequate model for the Cloud. Due to the many possible optimizations and performance gains Cloud storage is almost always bound to a certain data model. Many Cloud providers even report multiple data stores for different data models (nearly relational, key-value stores, unstructured large object stores). For example Large parts of Google's data are stored on GFS which has inherent support for replication. Structured data are stored in BigTable. Facebook's datamodel is graph-based consisting of Fbobj and associations among them. Salesforce's datamodel is essentially relational comprising relational tables and indices. In addition there several different data store types that emerge in Cloud environments. Key Value stores are very widely spread both as storage service and as caching tier. Almost every Cloud provider has a document store or a large object store. Researchers [37] unify those under the term NoSQL databases. Interestingly enough the optimizations and scalability depend indirectly on the application as well. For example both Salesforce and Facbook rely on SQL databases [38,33], however Salesfore can very well partition on tenant level (see multi-tenancy), while Facebook cannot due to the strongly referenced graph nature of their data.
- *Strong and Stable API*. The interface which Cloud frameworks exposes to the Cloud applications should be precisely controlled. In fact many SaaS vendors such as salesforce [33] claim to support internally a single version of their codebase, but emulate older versions. While this allows for a significant maintenance, development and performance gain, it imposes strict requirements on applications and ISVs.
- *Software Licensing*. Software licenses may pose a significant acceptance barrier to the IaaS model. It exposes directly elastic computational resources such as CPUs or storage, while allowing to deploy running images with pre-installed commercial software. Some software vendors require licenses not for present elastic resources, but rather for the physically available hardware resources on the server.

User Related

- *Parallelism*. Commodity hardware, clusters of 10 000 cores and 5 000 computers Cloud systems horizontally scalable with geographical partitioning. To make the best of this horizontal scale computing facility applications should be translatable to parallel jobs that can run onto all these machines in parallel. This is the only way to satisfy the high number of user requests per second against terabytes of data. Yet not every application and not every algorithm implemented in a certain programming language can be translated efficiently into small parallelizable tasks. The task complexity is minimized by employing special-purpose programming languages, especially for data processing ([39]). For example, Google's MapReduce [40] framework

and its open source implementation Hadoop by Yahoo [41] provides an excellent utility for Cloud data analysis. Yet complex requests such as ones employing joins cannot be implemented efficiently on MapReduce. Such issues are address by for instance Yahoo's Pig framework in which complex data analysis tasks are expressed in the Pig Latin language and compiled down to MapReduce jobs and deployed on Haddop. Hence not every enterprise application can make optimal use of the Cloud advantages without a complete re-design.

- *Data-Model*. Most of the enterprise applications operate on a relatively well-defined data model. Web- or Cloud applications, alternatively, must be able to handle both structured and unstructured, combine multimedia and semantic metadata, handle text natively. This poses the grand challenge of uniform and efficient handling. Cloud applications have to operate on a general data model delegating all of the heavy-lifting to the infrastructure. (See 2.3.1 and 2.3.1.4). In practice however different Cloud applications rely on different models hence the infrastructure has to support different sores to optimally handle them. For example, Facebook relies on a interconnected graph model comprising FbObjects and Associations, which is heavily metadata driven [38]. Mail attachments will be stored on a large object store, while mail messages will go to a different system. Through heavy use of metadata these pieces of information can be correlated properly. Important is that the infrastructure provides a way to combine services from different stores.

- *Strict ACID-Transactions*. Transactional guarantees are one of the key characteristics of enterprise applications. While ACID transactions represent a very powerful mechanism, it has been proven that they incur a significant performance overhead in highly distributed systems. Many authors argue that [42,43] in the Cloud availability should be traded for consistency. In addition models such as eventual consistency [43] have been proposed. The implications for the applications a manifold: (i) the application developer should custom-build consistency mechanisms; (ii) applications relying on consistency are no suitable for the Cloud. (iii) applications relying on strong consistency guarantees continue to exists on an enterprise facility are exposed a service and are consumed in the Cloud. This however is the least desirable alternative.

- *Security and Encryption*. While the majority of the Cloud providers report major security efforts as part of their infrastructure, security is widely viewed as a major hurdle towards Cloud migration. Enterprise systems with heightened security are designed to operate directly on encrypted data. Such an approach can be adopted in Cloud applications. On the one hand it entails significant restrictions on the applications feature set and design. On the other hand it leverages well with the abundant computing power available within the Cloud.

- *Interoperability*. While the predominant view nowadays is that applications are developed for a certain Cloud provider, this dependence seems too risky and is considered a hurdle for wide Cloud migration [10]. Out of this reason researchers recognize the need to be able to interoperate between Clouds and ideally be able to migrate. Given the experience for distributed computing this task will be very difficult.

3 Requirements of Data-Intensive Applications

The scale is the single most essential property that dominates the discussion of data-intensive applications in enterprise computing and in a Cloud environment. It influences many application characteristics such as basic assumptions, architecture, algorithms as well as features.

3.1 Types of Data-Intensive Applications

While the two general archetypes of data-intensive applications (OLAP and OLTP) are still present they differ from their classical meaning.

OLAP. On-Line Analytical Processing (OLAP) refers to querying mostly historical and multi-dimensional data with the purpose of analyzing it and reporting certain figures. The term is generalized by the terms business intelligence und analytics, although these carry a semantics of their own. In the Cloud most vendors offer approaches and technologies for performing analytics. These analyze terabytes of data calculating statistics or performing aggregate functions. Typical analytical operations such as relational joins are very difficult to implement. Other operators are also difficult to implement [44]. A major advantage and design goal is to be able to process TB of data, and therefore scale on a myriad of small inexpensive machines. This high degree of parallelism and elasticity requires compromises on the algorithmic richness and on the data model part. Nonetheless Cloud providers such as Salesforce do offer a close-to-relational data model and operator set. They have found an elegant way to bring the relational technology to the Cloud. Most of the analytical applications match the Cloud very well. They depend mostly on the cumulative bandwidth with which the data can be read and transferred - bandwidth is present within a Cloud data centre. They depend on the CPU power which is available as well.

OLTP. Although many Cloud applications rely on a batch update mode or expect rare updates there are many cases where frequent updates are rather the rule. Facebook and Salesforce are good example of that. Users of both systems update their enterprise data, profiles, pictures or status frequently. Although updates are a weakness in the Cloud environment both systems seem to handle them very well. The CAP theorem explains the key factor preventing Cloud systems from handling updates. On the one hand to increase the availability (and account for possible hardware failures) Cloud systems replicate data. If updates are to be handled consistently all replicas must be updated before the update operation is acknowledged, which is time consuming and blocks system resources. This cannot be done at the high request rate these systems have to serve. Therefore in the Cloud systems are optimised for availability not consistency [42,43]. In systems such as Facebook or Salesfore consistency is mostly achieved through cache invalidation or partitioning.

3.2 Characteristics Regarding Data

Data partitioning (or data sharding) is the ability to place data into buckets so that every bucket is updated independently of each other. For example, the data of a tenant is a

subset of the global tenant table that is updated by a certain set of clients. It is than possible to take that tenants data and place it in a data center in geographical proximity to the client. This is the mode chosen by the architects of Salesfore. Facebook's data on the other hand is very strongly connected. Therefore data partitioning cannot be considered realistic option. Constructing a page on Facebook requires data from different friends of a person. It is difficult to predict the exact geographical distribution of the friends. Even if the system manages to do so there is no guarantee that in future friends from different locations will join. Therefore it is impossible to partition data across Facebook datacenters. Therefore in absence of good partitioning possibilities a pulling friends status from a geographically remote center may slow down the construction of a page. Instead of partitioning [38] Facebook relies on metadata replication, main memory data processing, distributed indexing as well as multiple systems performing specializes data querying.

4 Discussion: Cloud vs. On-Premise

In an on-premise scenario systems with classical three tier architectures are very successfully utilised. Since these are located in the same data centre, latencies are low, throughput is high and the storage high performant. CRM as well as ERP systems are examples of OLTP systems. In a Cloud environment these work well only if the data can be partitioned/sharded properly to increase scalability. In a multi tenant environment it is needed to assign a tenant to a data centre near the client (or the majority of clients). Interestingly enough the tree tier architecture is still preserved within a data centre. Due to the fact that tenants are exclusively allotted to a data centre transactions do not span data centers thus being delayed by high latencies. Text processing on the other hand poses different requirements. In terms of transactions text processing implies long running transactions, where a document can be checked out, processed and checked back in. Thus it minimizes resource contention on shared resources, since no long locks are being held and consistency is being enforced upon check in. Interestingly this mode can be very well combined with versioning; for instance depending on the versioning semantics a new branch or a new version can be created. The check in of a document can trigger a whole set of post-processing operations (such as indexing, analytics ETL etc.) operating on the bulk of the document. Regardless of the rich client presentation requirements this mode of operations is very suitable for the Cloud. Moreover the transparent check-in/check-out operations are performed within the users' private workspace on the Cloud and not on the local machine (as with classical version control systems) thus avoiding possible bandwidth issues.

5 Related Work

By blending economic and technological aspects as well as blurring the line between application and infrastructure, the concept of Cloud Computing is subject to a wide spectrum of academic research. Hence a multitude of academic publications is available. We would like to point out some exemplarily samples we think most suitable for reflecting on the topic of Cloud Computing in its various dimensions.

Armbrust et al. analyse the phenomenon of Cloud Computing by starting from the hardware-level [10,35]. Foster et al. define the key issues of Grid-Computing in [15] and point out the main differences to Cloud Computing in [11]. Yang and Tate give a descriptive review of 58 academic articles regarding Cloud Computing as in 2009 [45]. Han surveys the market acceptance of Cloud Computing in [23]. Greenberg et al. look at possibilities to reduce costs in data centers from a provider related point of view in [18]. Church et al. discuss advantages and drawbacks of geo-diverse, distributed or containerized designs for data centers [19] .

The problem of the concurring requirements consistency, availability and performance in distributed systems is formalized as CAP-Theorem in [46] and further discussed in [43,47,48]. Bining et al. show the first steps towards a benchmark for Cloud solutions in [49]. There are multiple new technologies that are especially developed for Cloud scenarios. Some of the most prominent representatives cover the areas of structured data storage, analytics and caching. Google's BigTable [50] technology is one of the leading structured data stores. Alternative implementations are Yahoo! HBase [51], or Hypertable [52]. Facebook's Cassandra [53] is another distributed storage platform. Amazon S3 is another example of Cloud data store. In terms of analytics some of the dominating frameworks are Google's MapReduce [54] and its open source implementation from Yahoo Hadoop [41]. In addition there are numerous data caching technologies such as Facebook's MemCached [55].

In addition to the publications mentioned above, there is a nearly infinite number of unreviewed industry techreports, whitepapers and presentations available today. Due to the fact that Cloud Computing is mainly an industry-driven topic, we consider some of these to be noted when trying to understand and analyse the phenomena of Cloud Computing. Thus we would like to recommend a few of them [56,57,58,59,60,61].

6 Summary and Conclusions

In the present paper we analyzed different aspect of data intensive Cloud Computing applications and infrastructure. We analyzed requirements and the presented status quo between infrastructural and application requirements, between economic and technical factors. Cloud Computing on the one hand has a wider range of requirements to satisfy compared to enterprise computing, on the other hand many issues due to scale, elasticity and simplicity are still unsolved or implemented in a custom-taylored way. In addition Could Computing as a paradigm inherently offers significant diversity in terms of possible approaches (IaaS, Paas, SaaS, etc.) and possible architectural and design alternatives.

We also reach the conclusion that applications and Cloud infrastructures are tightly coupled and influence each other. In order to achieve high performance and cope with issues resulting from the large scale many Cloud vendors resort to custom solutions. New algorithms and paradigms are developed from scratch (e.g. Google's MapReduce). In this respect data sharding, transactions consistency and isolation are some of the dominating issues. On the other hand many Cloud providers aim at reusing existing approaches and technologies from the field of enterprise computing. One of the issues looming on the horizon is the mechanisms for designing Cloud applications and infrastructures.

To recapitulate data-intensive Cloud Computing is a very fast evolving field offering much room for innovation. New approaches can be expected in the field of transaction processing, access paths, architectures, caching and analytics.

References

1. YouTube: Youtube - broadcast yourself (2010), http://www.youtube.com/
2. Yahoo!: Flickr (2010), http://www.flickr.com
3. Ebay: ebay - new & used electronics, cars, apparel, collectibles, sporting goods & more at low prices (2010), http://www.ebay.com
4. Google: Google picasa (2010), http://www.google.com/picasa
5. Microsoft: Microsoft hotmail (2010), http://www.hotmail.com
6. Amazon.com: Online shopping for electronics, apparel, computers, books, dvds & more (2010), http://www.amazon.com
7. Google: Google documents and spreadsheets (2010), http://www.google.com/docs/
8. Facebook: Facebook (2010), http://www.facebook.com
9. Google: Google maps (2010), http://maps.google.com/
10. Armbrust, M., Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., et al.: A view of cloud computing. Communications of the ACM 53(4), 50–58 (2010)
11. Foster, I., Zhao, Y., Raicu, I., Lu, S.: Cloud computing and grid computing 360-degree compared. In: Grid Computing Environments Workshop, 2008. GCE 2008, pp. 1–10 (2008)
12. Mell, P., Grance, T.: The NIST Definition of Cloud Computing. National Institute of Standards and Technology, Information Technology Laboratory (July 2009)
13. Bittman, T.: A better cloud computing analogy (September 2009), http://blogs.gartner.com/thomas_bittman/2009/98/22/a-better-cloud-computing-analogy/
14. Giordanelli, R., Mastroianni, C.: The cloud computing paradigm: Characteristics, opportunities and research issues. Technical Report RT-ICAR-CS-10-01, Consiglio Nazionale delle Ricerche Istituto di Calcolo e Reti ad Alte Pestazioni (April 2010)
15. Foster, I.: What is the grid? A three point checklist. GRID Today 1(6), 22–25 (2002)
16. Mäkilä, T., Järvi, A., Rönkkö, M., Nissilä, J.: How to define software-as-a-service - an empirical study of finnish saas providers. In: Tyrväinen, P. (ed.) ICSOB 2010. Lecture Notes in Business Information Processing, vol. 51, pp. 115–124. Springer, Heidelberg (2010)
17. Ross, J.W., Westerman, G.: Preparing for utility computing: The role of it architecture and relationship management. IBM Systems Journal 43(1), 5–19 (2004)
18. Greenberg, A., Hamilton, J., Maltz, D.A., Patel, P.: The cost of a cloud: research problems in data center networks. SIGCOMM Comput. Commun. Rev. 39(1), 68–73 (2009)
19. Church, K., Greenberg, A., Hamilton, J.: On delivering embarrassingly distributed cloud services. Hotnets VII (2008)
20. Patel, C.D., Shah, A.J.: Cost model for planning, development and operation of a data center. Technical report, HP Laboratories Palo Alto (June 2005)
21. Banks, D., Erickson, J., Rhodes, M.: Multi-tenancy in cloud-based collaboration services. Technical Report HPL-2009-17, HP Laboratories (February 2009)
22. Grossman, R.L.: The case for cloud computing. IT Professional 11(2), 23–27 (2009)
23. Han, L.: Market Acceptance of Cloud Computing - An Analysis of Market Structure, Price Models and Service Requirements. Bayreuth Reports on Information Systems Management, p. 42 Universität Bayreuth (April 2009)

24. Varia, J.: Architecting for the cloud: Best practices (January 2010), http://jineshvaria.s3.amazonaws.com/public/cloudbestpractices-jvaria.pdf
25. Amazon.com: Amazon web services (2010), http://aws.amazon.com
26. Reese, G.: Cloud Application Architectures: Transactional Systems for EC2 and Beyond, 1st edn. O'Reilly, Sebastopol (2009)
27. Google: What is google app engine (2010), http://code.google.com/intl/en/appengine/docs/whatisgoogleappengine.html
28. Google: Google spreadsheets api (2010), http://code.google.com/intl/en/apis/spreadsheets/
29. Google: Google document list api (2010), http://code.google.com/intl/en/apis/documents/
30. Google: Gmail apis and tools (2010), http://code.google.com/intl/en/apis/gmail/
31. Microsoft: Windows Azure (2010), http://www.microsoft.com/windowsazure/windowsazure/
32. Microsoft: SQL Azure - database as a service (2010), http://www.microsoft.com/windowsazure/sqlazure/
33. Woollen, R.: The internal design of salesforce.com's multi-tenant architecture. In: Proceedings of the 1st ACM symposium on Cloud computing, SoCC 2010, pp. 161–161. ACM, New York (2010)
34. Schroeder, B., Pinheiro, E., Weber, W.D.: Dram errors in the wild: a large-scale field study. In: Proceedings of the eleventh International Joint Conference on Measurement and Modeling of Computer Systems, SIGMETRICS 2009, pp. 193–204. ACM Press, New York (2009)
35. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M.: Above the clouds: A berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley (February 2009)
36. Appel, S.: Analysis and Modeling of Application Behavior in Virtualized Environments. Master's thesis, Technische Universität Darmstadt (2009)
37. Stonebraker, M.: Sql databases v. nosql databases. ACM Commun. 53(4), 10–11 (2010)
38. Sobel, J.: Building facebook: performance at massive scale. In: Proceedings of the 1st ACM symposium on Cloud computing, SoCC 2010, pp. 87–87. ACM, New York (2010)
39. Olston, C., Reed, B., Srivastava, U., Kumar, R., Tomkins, A.: Pig latin: a not-so-foreign language for data processing. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, pp. 1099–1110. ACM Press, New York (2008)
40. Dean, J., Ghemawat, S.: Mapreduce: Simplified data processing on large clusters. In: Proceedings of the Sixth Symposium on Operating Systems Design and Implementation, OSDI 2004, pp. 137–150 (December 2004)
41. Yahoo!: The hadoop project (2010), http://hadoop.apache.org/core/
42. Brantner, M., Florescu, D., Graf, D., Kossmann, D., Kraska, T.: Building a database on s3. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, pp. 251–264. ACM Press, New York (2008)
43. Vogels, B.Y.W.: Eventually consistent. Communications of the ACM 52(1), 40–44 (2009)
44. Stonebraker, M., Abadi, D., DeWitt, D.J., Madden, S., Paulson, E., Pavlo, A., Rasin, A.: Mapreduce and parallel dbmss: friends or foes? ACM Commun. 53(1), 64–71 (2010)
45. Yang, H., Tate, M.: Where are we at with cloud computing? In: Proceedings of the 20th Australasian Conference on Information Systems, ACIS 2009, pp. 807–819 (2009)
46. Gilbert, S., Lynch, N.: Brewer's conjecture and the feasibility of consistent available partition-tolerant web services. ACM SIGACT News 33(2), 51–59 (2002)

47. Finkelstein, S., Brendle, R., Jacobs, D.: Principles for inconsistency. In: Proceedings of the 4th Biennial Conf. on Innovative Data Systems Research (CIDR), Asilomar, CA, USA (2009)
48. Brown, A.B., Patterson, D.A.: Embracing failure: A case for recovery-oriented computing (roc). In: High Performance Transaction Processing Symposium, vol. 10, pp. 3–8 (2001)
49. Binnig, C., Kossmann, D., Kraska, T., Loesing, S.: How is the weather tomorrow? towards a benchmark for the cloud. In: Proceedings of the Second International Workshop on Testing Database Systems (DBTest 2009). ACM, New York (2009)
50. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: a distributed storage system for structured data. In: Proceedings of the 7th symposium on Operating Systems Design and Implementation, OSDI 2006, pp. 205–218. USENIX Association, Berkeley (2006)
51. Team, H.D.: Hbase: Bigtable-like structured storage for hadoop hdfs (2007), `http://wiki.apache.org/lucene-hadoop/Hbase`
52. Hypertable.org: Hypertable (2010), `http://hypertable.org/`
53. Lakshman, A., Malik, P.: Cassandra: a decentralized structured storage system. SIGOPS Oper. Syst. Rev. 44(2), 35–40 (2010)
54. Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. In: Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation, OSDI 2004, pp. 10–10. USENIX Association, Berkeley (2004)
55. Project, M.: What is memcached? (2010), `http://memcached.org/`
56. Fenn, J., Raskino, M., Gammage, B.: Gartner's hype cycle special report for 2009 (2009)
57. Dubey, A., Mohiuddin, J., Baijal, A.: Emerging Platform Wars in Enterprise Software. Technical report, McKinsey & Company (2008)
58. Dubey, A., Mohiuddin, J., Baijal, A.: Enterprise Software Customer Survey 2008. Customer survey, McKinsey & Company, SandHill Group (2008)
59. Gens, F.: Top 10 predictions: Idc predictions 2010: Recovery and transformation. Survey, IDC (December 2009)
60. Hagiu, A., Yoffie, D.B.: What's your google strategy? Harvard Business Review, 74–81 (April 2009)
61. Thethi, J.P.: Realizing the value proposition of cloud computing. Technical report, Infosys (April 2009)