HYBRID CLOUD ARCHITECTURE FOR SHORT MESSAGE SERVICES

Yrjo Raivio[1], Oleksiy Mazhelis[2], Koushik Annapureddy[1],
Ramasivakarthik Mallavarapu[1] and Pasi Tyrväinen[2]
[1]*Department of Computer Science and Engineering, Aalto University, Aalto, Finland*
[2]*Department of Computer Science and Information Systems, University of Jyväskylä, Jyväskylä, Finland*

Keywords: Hybrid Cloud, Short Message Service, Cost Evaluation, Traffic Characterization, Autoscaling.

Abstract: Dedicated and expensive computing platforms are commonly applied to mobile network systems. This is necessary, despite the economic burden, due to strict performance requirements in availability, latency and throughput. However, cloud computing is changing the rules of the game by offering cost efficient and high performance computer systems. Pay-per-use principle is helping network administrators to scale the computing capacity on a need basis, reducing both capex and opex costs. Several networks can benefit from this advantage in wireless services, including both end user and internal back end services. The focus in this paper is on the Short Message Service (SMS), which is one of the most successful and widespread end user services after voice in mobile networks. The SMS Center (SMSC) is used as a test bed to optimize the usage of public and private clouds in network operations, both in technology and business. This paper presents a hybrid cloud architecture that enables an automatic up-and-down-scaling of the system, using dynamic resource provisioning and depending on the service load. In addition, a cost analysis to find the optimal balance between public and private clouds is described. Finally, the proposed solution is thoroughly evaluated, future research ideas are highlighted and conclusions are drawn.

1 INTRODUCTION

It is a common belief (Murphy, 2010) that cloud computing cannot yet offer the performance level that is required in carrier grade mobile services. This is partly true for real time services such as voice. On the other hand, most other mobile operations can be executed in clouds (Gabrielsson et al., 2010). Operators already offer cloud computing services to their customers in fact, but until now, very few of them have transferred their own operations to cloud. However, the situation is changing rapidly, and vendors in particular are evaluating opportunities in the cloud space. The easiest options for applying clouds in the telecom industry can be found in back end systems such as business support systems (BSS) (Raivio and Dave, 2011) and operational support systems (OSS), but also the core services, including database (Paivarinta and Raivio, 2011) or voice control systems (Venugopal et al., 2011), can be implemented in clouds.

A Short Message Service (SMS) is a good example of a service that can be run in a cloud. Text messaging has been a globally very successful end user service. Roaming agreements and standardized technologies have guaranteed smooth interoperability over operator borders, offering a good service experience for end users and economic success for operators. Although the SMS growth rate has stabilized in the developed markets, the message volumes are still increasing in the developing countries, and for that reason system and cost optimizations are sought after.

Telecommunication traffic fluctuates, and the SMS traffic is no exception. During daytime and exceptional events or incidents, messages are sent much more frequently than in the night. In current SMS systems, the capacity is designed on the basis of the peak load and computation resources are wasted during silent periods due to the dedicated nature of the systems. Cloud computing can significantly improve the situation because virtual machines (VM) located in the public cloud can be turned off when computing power is not anymore required and vice versa (Khajeh-Hosseini et al., 2010). On the other hand, ex-

Raivio Y., Mazhelis O., Annapureddy K., Mallavarapu R. and Tyrväinen P..
HYBRID CLOUD ARCHITECTURE FOR SHORT MESSAGE SERVICES.
DOI: 10.5220/0003908104890500
In *Proceedings of the 2nd International Conference on Cloud Computing and Services Science* (CLOSER-2012), pages 489-500
ISBN: 978-989-8565-05-1

tensive usage of the public cloud can become expensive. Furthermore, operators must support high availability and unconditional data security, and for those reasons operators are forced to seek compromises.

One alternative to solve the described challenges is called a hybrid cloud. The main idea here is to utilize private and public clouds in parallel. In this setup, the private cloud provides the base capacity, while the public cloud can be used for optimal load balancing. However, finding an optimal setup is not an easy task. The solution should take into account traffic variations, and based on those, tune the system to provide simultaneously high performance and low cost. This paper focuses on studying the applicability and feasibility of using the hybrid cloud in the context of telecommunication services, particularly the SMS case.

The paper is organized as follows. Section 2 introduces the background and related work of the research along with a short review of SMS architecture, hybrid cloud and dynamic resource provisioning model. Section 3 describes in detail the proof of concept of the SMS service in a cloud. Section 4 elaborates the cost analysis, giving reasons for the selection of the optimal hybrid architecture. Next, Section 5 critically evaluates the results using both reactive and proactive resource provisioning models, and Section 6 summarizes the work and proposes future research ideas. Finally, Section 7 offers paper's conclusions.

2 BACKGROUND AND RELATED WORK

This section explains the basics and related work of the SMS architecture, SMS traffic characteristics, the hybrid cloud concept and also the autoscaling algorithms utilized in the verification of the results.

2.1 SMSC Architecture and Load

The key element of the text messaging system in mobile networks is the Short Message Service Center (SMSC). It is connected to the Mobile Switching Center (MSC), the Visitor Location Register (VLR), the Home Location Register (HLR), and further to the application servers (AS) and the other SMSCs. Text messages are first sent to the closest MSC, which locates the home SMSC where the message is to be forwarded. Depending on the content and address of the message, it is sent either to the next SMSC, to the AS or - if the receiver belongs to the same SMSC domain - directly to the receiver. Basically SMSC acts as a store-and-forward router. If the SMSC is overloaded, the MSC will store the messages and will resend them to the SMSC later. In the case when the message receiver suddenly is outside the network, the SMSC will store the message and send it to the receiver the moment it joins the network again. The SMSC architecture is presented in Figure 1 (Mouly and Pautet, 1992).

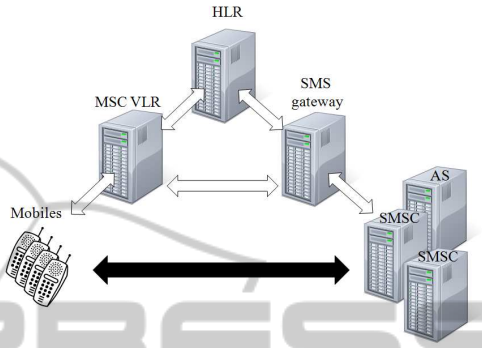

Figure 1: SMSC architecture.

The SMSC load varies depending on the time of day. During weekdays the load is normally very periodic, but at the weekend the load curve slightly changes. Figure 2 shows an example of the SMS load curve during a full week, starting from Monday and ending on Sunday. The data has been collected from an SMSC located on the network of an European operator. The curve does not include the absolute volume, but for this research the knowledge of data volume dynamics is enough. The message volume can be extrapolated for various markets when the total message volume of the market is known.

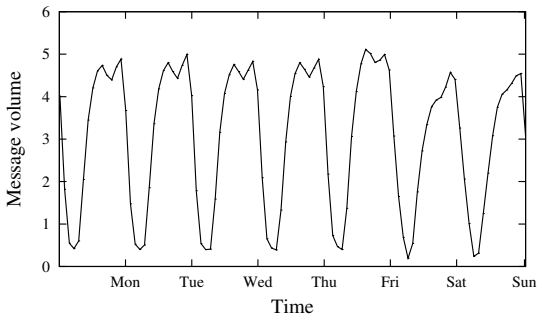

Figure 2: Traffic distribution during one week.

It is notable that the example trace does not show any exceptional traffic peaks that might occur during accidents, special events and certain dates, such as New Year's Eve. It is estimated that the message peaks ultimately can be as much as up to tenfold compared to the usual base load (Zerfos et al., 2006). However, due to competitive communication methods, text messaging peaks are not so common

anymore as they used to be before the introduction of social networks and mobile email.

For evaluating the proof of concept and cost, the real numbers for the load curve shown in Figure 2 should be added. Using the scale 0 to 6, the average of the volume during the whole week is 3.07. According to ITU-T[1] it was estimated that in 2010 6 trillion SMS messages were sent. That number equals a continuous load of slightly less than 200 000 messages per second. On the other hand, an average mobile user sends approximately 1000 messages per year[2], which fact means that a market with 10 million subscribers typically sends 10 billion SMS messages in a year. If the numbers are applied to our example, we may set up the value 1 to equal a rate of 100 messages per second, which yields on average a rate of 307 messages per second during a week and about 10 billion messages per year.

2.2 Hybrid Cloud

In the current SMSC systems, operators use dedicated hardware and software solutions. The capacity of the systems is tailored according to the maximum load, but peak loads are usually excluded from the calculations. In the case of a sudden peak occurring, SMSCs become overloaded and service quality deteriorates. It means that the message delivery time will become longer due to retransmissions between the MSCs and SMSCs. Scaling the SMSC capacity on the basis of a rare peak load does not make sense either, because it would mean high and wasted capex costs.

Cloud computing can help to design a system that automatically scales up and down depending on the load. Public clouds support autoscaling functions, and the pay-per-use principle will reduce the capex costs. Given this background, a logical solution would be to set up the whole system in a public cloud. However, in the long run the public cloud CPU hourly cost is more expensive than the same hour in the private cloud (Walker, 2009). It is also mathematically proved that the hybrid cloud is the most economical solution upon variable traffic load (Weinman, 2011).

The basic idea with a hybrid solution is simple. A service provider invests in a private cloud that can manage the performance critical applications, while a public cloud can run background processes (Hajjat et al., 2010). However, our research focuses on a dynamic hybrid cloud approach. In case a peak load occurs, all excess traffic is forwarded to VMs located in the public cloud, while the VMs in the private cloud deal with the base load. When the total load again drops below the capacity of the private cloud, the public VMs are shut down. The system also has to have an autoscaling functionality that can make correct decisions on the load balancing between private and public clouds, and which can also control the VMs in the public cloud (Moreno-Vozmediano et al., 2009).

The costs of a hybrid cloud are comprised of the costs of the private and the public portions of the cloud. Private cloud costs are incurred due to acquiring, deploying and operating private cloud hardware and software, and therefore it depends on the volume of computing capacity acquired for the private cloud. Public cloud capacity is charged on the basis of the actual usage, and hence the public cloud cost depends on the length of time when the public cloud capacity is used.

Both the private and the public cloud cost depend on how the demand for computing capacity is divided between the pre-acquired infrastructure (private cloud) and the consumed on-demand infrastructure (public cloud). Several approaches are available for cost-optimally distributing the load within cloud (Martens and Teuteberg, 2011) and within a hybrid cloud, e.g. by optimally allocating computing tasks to cloud resources at each time slot (Strebel and Stage, 2010), or by determining the fixed cost-optimal threshold for the private cloud resources, and allocating the excess load to the public cloud (Mazhelis and Tyrväinen, 2011).

2.3 Dynamic Resource Provisioning

Performance unpredictability is one of the major challenges of cloud computing (Armbrust et al., 2010). Autoscaling in the hybrid cloud is an important enabler for a high availability and a carrier grade performance. Due the relatively long startup time of the public cloud VMs, the system must launch new instances well before they are required. However, designing an automated resource controller is a complex task. It is essential that we can accurately estimate the workload handling capacity of an individual VM in both private and public IaaS cloud environments. We broadly categorize the resource controllers into reactive and proactive models.

In addition to load monitoring, resource provisioning can be based on cost-efficiency and some other parameter, such as the deadline of application execution (Mao and Humphrey, 2011; Van den Bossche et al., 2010), or performance (Chang et al., 2010). Compared to our research setup, the previous ex-

[1] http://www.itu.int/ITU-D/ict/material/FactsFigures2010.pdf

[2] http://epp.eurostat.ec.europa.eu/statistics_explained/index.php/Telecommunication_statistics

amples include a set of interconnected applications, while our work concentrates on analyzing a single application that includes several components. The result is slightly different if the optimization is reviewed from a service provider's (SP) point of view. One paper (Mazzucco et al., 2010) evaluates how the different scaling strategies bring the best profit for the SP, and at the same time minimizes the energy consumption. The energy part could be applied to the user of the service, as well.

2.3.1 Reactive Model

In the reactive model the resource controller constantly monitors the incoming message traffic and scales up the resources once a substantial increase in the workload is detected. This approach works well in theory, but in practice, especially during unpredictable or dynamically varying workload peaks, it fails to maintain the required Quality of Experience (QoE). The reactive model suffers from the fact that the resource controller will start the process of resource provisioning only when an increase in the load is detected, even though it is known that a VM requires at least 2 to 3 minutes to reach an operational state. During the startup time we observed a steep increase in the message processing time, a fact which will create a heavy loss in QoE.

2.3.2 Proactive Model

Dynamic resource allocation schemes that adapt to varying workloads will involve three phases at a minimum: extensive monitoring of incoming message traffic, workload modeling and subsequent decision making on scaling resources. Proactive models, especially, need to characterize the incoming workload, predict the next few steps and act according to the prediction preemptively. Fortunately there are several predictive models in use in various fields like econometrics, machine learning and neural networks.

Research on using proactive models for autoscaling in cloud computing is a hot topic, and interesting results have been recently published. For example, linear regression is applied to solve optimal cost parameters for deploying business software to hybrid cloud (Strebel and Stage, 2010). Probabilistic model checker called PRISM evaluates the performance of VM live migration (Kikuchi and Matsumoto, 2011). The PRESS algorithm (Gong et al., 2010) uses signal processing and statistical methods to identify repeating patterns and discrete-time Markov chain to predict load in the near future. Markov models suit well to analyze http traffic that is often heavy-tail type (Pacheco-Sanchez et al., 2011). Finally, statistical methods such as autoregressive moving average (ARMA) (Roy et al., 2011) supported by Fourier Transfer analysis (Zhang et al., 2009) are highly efficient in modeling stationary time series data.

The SMS traffic fulfills the stationary characteristics, and for that reason we chose the ARMA model (Box and Jenkins, 1990) for predicting the future workload based on the recent history of the workload pattern. The resource controller forecasts incoming message traffic every minute, and it is assumed to be centralized and not distributed like in (Yazir et al., 2010; Ardagna et al., 2011). We used a first order ARMA filter to model and predict the incoming message traffic rates. This filter is represented by the following equations:

$$X(t) = \mu + \alpha \times X(t-1) + \beta \times \varepsilon(t-1) + \varepsilon(t) \quad (1)$$

$$\mu = Mean \times (1-\alpha) \quad (2)$$

where

- μ is a constant;
- α and β are ARMA parameters, having values 0.90 and 0.15 respectively; and
- ε represents white noise.

In the experiments on Amazon EC2 Large that located in the EU region, we observed that the time required to provision a VM varies between 80 to 90 seconds. Taking this observation into account, we have defined our prediction interval as 120 seconds. The results with the workload prediction have been highly successful. Figure 3 shows the predicted workload compared with the actual workload for one weekday, using Figure 2 as input material. The results were good, presenting in average 4.5% prediction error.

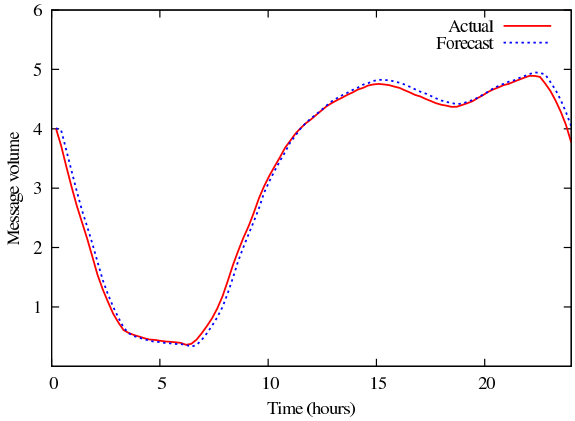

Figure 3: Workload prediction for one day.

3 PROOF OF CONCEPT

This section describes the experimental system and algorithms of the resource controller in detail.

3.1 Experimental System

An experimental setup was created to review the functionality of the hybrid cloud. In order to simulate the features of the SMSC in a VM, an SMSC simulator was bundled as a black box into a VM image. The SMSC functionality was offered by a simulator called SMPPSim[3], which uses the Short Message Peer-to-Peer (SMPP) protocol for communication. The variable message load was created by OpenSMPP API[4]. Next, the threshold capacity value for a VM was measured. The measurement results using one EC2 Large instance are shown in Figure 4. The results indicate that the time required for processing a message increases dramatically for a workload above 100 requests per second. Each time there was a substantial increase in the workload, QoE worsened due to the ineffective reactive resource provisioning approach.

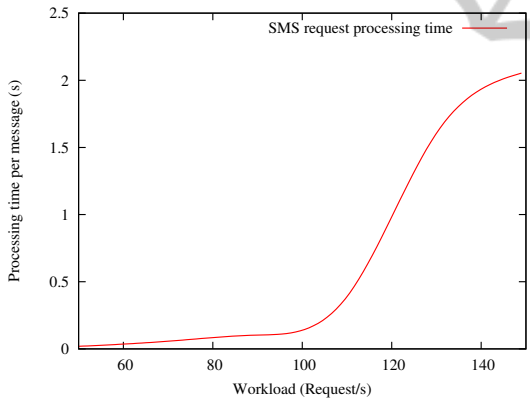

Figure 4: SMS message processing time with EC2 Large.

The optimal hybrid solution can be calculated when the capacity and CPU hourly costs of the private and public clouds are known. In the setup, Amazon EC2[5] was chosen for the public cloud provider due to the rich feature set, while the private cloud component was taken from OpenNebula[6], which offers a superior openness as well as good integration capabilities with several public clouds. The private cloud includes a *Load Balancer* that is implemented using an open source software called HAProxy[7]. The Load Balancer can distribute the SMPP messages to the underlying VMs located both in the private cloud and public cloud.

The *Resource Controller* includes two blocks: the *Monitor* software based on the TCPSTAT[8], and an optional *Forecast* functionality. The usage of the Forecast block is described in the following subsection. The Monitor reports periodically the load detected by the Load Balancer to the *Resource Allocator* that is implemented on the XML-RPC[9] client. Based on the received information, the Resource Allocator, help from the *Virtual Infrastructure Manager*, scales up or down the resources available to the system. A virtual bridge inside the private cloud enables communication between the SMSCs, while on the other hand, the Load Balancer provides a gateway to the public network. Figure 5 presents the architecture for the experimental setup.

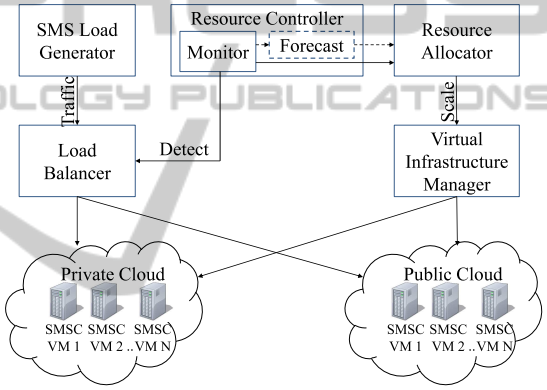

Figure 5: Hybrid cloud architecture for SMSC.

Initially VMs are launched in the private cloud until the overall maximum capacity is achieved. When that happens, the front end, implemented on the OpenNebula platform, starts new VMs in the public cloud, thus forming the hybrid cloud. If there is a need to scale the resources down, the front end shuts down the VMs in the public cloud. Before the SMSC is shutdown in the EC2, the undelivered messages are sent forward or saved to the private cloud. Due to privacy issues and government regulations, it is suggested that VMs are launched only in regions that meet the requirements of the authorities (Kaufman, 2009).

3.2 Algorithm

In order to improve the performance and increase the high availability of the SMSCs, a framework is de-

[3]http://www.seleniumsoftware.com/
[4]http://sourceforge.net/projects/smstools/
[5]http://aws.amazon.com/ec2/
[6]http://opennebula.org/
[7]http://haproxy.1wt.eu/
[8]http://www.frenchfries.net/paul/tcpstat/
[9]http://www.xmlrpc.com/

signed by assuming the capacity of an SMSC simulator. In this case, we know that an SMSC simulator can process up to λ_{inst} messages per second within a given service time. If there is a flow of more than λ_{inst}, then there would be a delay introduced by the simulators, thus increasing the service time for each message. Algorithm 1 explains the implementation details of the *Resource Controller* and Algorithm 2 describes the functionality of the *Forecast* module.

Algorithm 1 initially starts a fixed number of VMs in a private cloud. The monitoring management system tracks the load for every *k* seconds. Depending on the recent history of the workload, the *Forecast* module predicts the future workload. The module utilizes the ARMA prediction algorithm described in Subsection 2.3.2. In the first step we must stationarize the non-stationary time series to stationary one. The output is used as the input to the algorithm, and the resulting value is de-stationarized to obtain the predicted workload for the next prediction interval. Depending on the result, the required and also optimum number of VMs are deduced using the *map* function. Table 1 describes the variables used in the algorithms.

Algorithm 1: Resource Controller.

1: *start* VM in C_{pr}
2: $VM_{cur} \leftarrow VM_{pr}$
3: **while** *true* **do**
4: *monitor* $W_a(n)$;
5: $W_{ft}(n+1) \leftarrow Forecast(W_a(n))$;
6: $VM_\lambda \leftarrow map(W_{ft}(n+1))$;
7: **if** $VM_{cur} < VM_\lambda$ **then**
8: *scale up*;
9: **else if** $VM_{cur} > VM_\lambda$ **then**
10: *scale down*;
11: **end if**
12: *sleep* k;
13: **end while**

Algorithm 2: Forecast.

1: $W_s(n) \leftarrow stationarize(W_a(n))$;
2: $\mu = M \times (1-\alpha)$;
3: $\varepsilon(n) = W_{at}(n-1) - \beta \times \varepsilon(n-1)$;
4: $W_{st}(n+1) = \mu + \alpha \times W_{st}(n) + \beta \times \varepsilon(n)$;
5: $W_{ft}(n+1) \leftarrow destationarize(W_{st}(n+1))$;

4 COST ANALYSIS

Several approaches are available for cost-optimally distributing the load within a hybrid cloud, e.g. by optimally allocating computing tasks to cloud resources at each time slot (Strebel and Stage, 2010), or by determining the fixed cost-optimal threshold for the private cloud resources, and allocating the excess load to the public cloud (Mazhelis and Tyrväinen, 2011). In this section, we apply the last approach, which is more lightweight and whose outcomes are easier to interpret, in order to determine the cost-optimal division of the SMSC's computing capacity between the private and the public portions of a hybrid cloud.

Table 1: List of variables used in Algorithms 1 and 2.

Variables	Description
$W_a(n)$	Actual workload till nth instant
$W_{at}(n)$	Actual workload at nth instant
$W_{st}(n)$	Stationarized workload at nth instant
$W_{ft}(n+1)$	Forecasted workload at $(n+1)^{th}$ instant
C_{pr}	Private cloud
C_{pu}	Public cloud
map	Returns VM_λ
VM_{cur}	Number of running VMs
VM_{pr}	Fixed number of running VMs in Private cloud
VM_λ	Optimum number of VMs
k	Monitoring interval
$W_s(n)$	Stationarized actual workload values
α	AR coefficient
β	MA coefficient
μ	ARMA constant
ε	White Noise

4.1 Seeking Cost-efficient Division of Load in Hybrid Cloud

The unit of the above division of load is a *computing instance* that corresponds to a processor core (in the private cloud) or to an Amazon Large EC2 instance (in the public cloud). For the sake of simplifying the analysis, the cores are configured to approximately match the performance characteristics of the EC2 Large instances.

When the SMS request arrival rate grows, the instances in the private portion of the hybrid cloud are allocated first. In the event that all of the available private instances are already allocated, the public instances are launched. Alternatively, when the SMS request arrival rate decreases, the public instances are released first, after which the private instances are freed. Note that the decommissioning of the unused public instances is done at the end of the charging period (i.e., in case of EC2, at the end of an hour)

to avoid extra charging if two or more bursts occur within the same charging period.

Given the maximum SMS request arrival rate λ_{max}, the maximum number of required instances can be estimated as:

$$n_{max} = \lceil \frac{\lambda_{max}}{\lambda_{inst}} \rceil. \quad (3)$$

Denote the number of acquired private instances as n_{pr}. Then, the threshold portion of SMS requests processed by the private cloud is

$$q = \frac{n_{pr}}{n_{max}}. \quad (4)$$

While the SMS request arrival does not exceed the threshold

$$\lambda_T = q n_{max} \lambda_{inst} \quad (5)$$

these requests are processed by the private instances. When the arrival rate grows beyond λ_T, the excessive requests are processed using public instances.

Thus, the costs of the private and the public portions of the cloud depend on the threshold q. We will therefore look for the value of q, minimizing the overall costs: $q_{min} = \min_q C$.

According to the literature (Mazhelis and Tyrväinen, 2011), the cost-optimal time of using the public cloud $\tau_c(q_{min})$ can be estimated as

$$\tau_c(q_{min}) = \frac{(p_{co} + p_{bo}k)T}{u_c p_{co} + p_{bo}k(u_b + \rho(1+u_b))}, \quad (6)$$

where

- p_{co} is the unit price of the private cloud computing capacity;
- p_{bo} is the unit price of the private cloud data transfer;
- $u_c = p_{cc}/p_{co}$ (where p_{cc} is the unit price of the public cloud computing capacity) is the premium charged by the public cloud provider for the use of its on-demand computing capacity;
- $u_b = p_{bc}/p_{bo}$ (where p_{bc} is the unit price of the public cloud data transfer) is is the premium charged by the public cloud provider for the use of its data communications capacity;
- k is the volume of the data transferred between the hybrid cloud and the clients during the time a single unit of the computing capacity is consumed;
- ρ is the ratio of the data transferred between the private and the public portions of the cloud to the data transferred between the public cloud and the clients; and
- T is the overall time during which the costs are estimated (e.g. the depreciation period).

Eq. 6 indicates that the cost-optimal time of using the public cloud decreases i) as the premium charged by the cloud provider (u_c and u_b) grows, and ii) as the intensity of data communications (k and ρ) increases.

As was shown in (Mazhelis and Tyrväinen, 2011), τ_c is a decreasing function of q, and therefore there exists an inverse function $q_{min} = Q(\tau_c)$ that can be used to obtain the value of q_{min}. Thus, having found the value of τ_c, and using the knowledge of the SMS request distribution to estimate $Q(\tau_c)$, the value of q_{min} can be determined.

4.2 Cost Assessment

As discussed above, the hybrid cloud cost is the sum of the costs of the private cloud and the public cloud portions. The private cloud cost is independent of the rate of requests at each instant; rather, the cost is determined by the number of virtual cores allocated (i.e. acquired or leased) for the purpose of serving as an SMSC. The cost of the public cloud, on the other hand, depends on the volume of consumed capacity (CPU-hours, transferred data, etc.).

The cost of the private portion of the hybrid cloud encompasses the costs of acquiring hardware and software, as well as the costs of rack space, power, allocated bandwidth, and the technical support costs. These costs are estimated based on the pricing for virtual servers offered by a hosting service enterprise Nebula[10]. Specifically, the cost of a computing instance running Linux Debian5 OS, 2 GB of memory and a 120 GB hard disk supporting automated backups and high availability is estimated to be $C_A = \$233$ per month.

The costs accounted for in the public portion of the hybrid cloud include (based on the Amazon EC2 pricing (Ireland) for on-demand instances, effective on November 7, 2011):

- Consumption-based cost of on-demand computing capacity (no reservation is assumed) $C_L = \$0.38$ per hour per instance;
- Volume-based data transfer costs assuming $V_{IN} =$ 1680B and $V_{OUT} =$ 1322B of data transmitted per SMS request, respectively, into and out of the public cloud: $C_{IN} = \$0$ and (since volume is below 10TB per month) $C_{OUT} = \$0.12$ per GB;
- Volume-based storage costs (200 bytes per request, \$0.11 per GB);
- Volume-based costs of I/O operations (4 I/O per request, \$0.11 per million I/O requests);

[10] http://www.nebula.fi/

- Additional costs incurred due to the use of Elastic IPs, Load Balancing and CloudWatch.

In the real SMSC system, successfully delivered SMS messages are usually not stored. According to (Zerfos et al., 2006), 73% of SMS messages are delivered successfully within 10 seconds, implying that 27% of the messages should be stored for resending until they are successfully delivered. However, most of the messages are delivered successfully within a few minutes, which means that the volume-based storage and I/O operation requirements related to the message deliveries are very minor. The same comment applies to the storage costs of the VMs. The size of the VM is on the level of 100 GBs, resulting in only a price of $11 per VM in a month.

Likewise, the allocation of Elastic IPs with the price of $0.01 per hour incurs a minor yearly cost below $90 per IP. Load balancing or CloudWatch monitoring services will not be needed because the load balancing function was implemented outside of the EC2 domain and because the monitoring software was provided by the open source components. Therefore, only the cost of on-demand computing capacity and the data transfer costs will be taken into account when estimating the public cloud costs, whereas the other costs are assumed minor or absent and hence are ignored.

Finally, the maximum SMS arrival rate to be handled by an instance was estimated empirically by experimenting with the SMSC simulator. It was found that, should the SMS arrival rate grow above 100 messages per second, the time of handling a message would exceed the acceptable limit. Therefore, the maximum SMS arrival rate per core is assigned the value of $\lambda_{\text{inst}} = 100$.

Based on the assumptions and arguments above, the pricing parameters can be estimated as follows.

The price of private computing capacity is estimated per core per hour as:

$$p_{\text{co}} = \frac{C_{\text{A}}}{24 \times 30} = 0.32\,\$/\text{hour}. \tag{7}$$

The price of public computing capacity is estimated based on the Amazon EC2 pricing as:

$$p_{\text{cc}} = 0.38\,\$/\text{hour}. \tag{8}$$

Furthermore, based on (Hamilton, 2010), we can assign

$$p_{\text{bo}} = 0.04\,\$/\text{GB}, \tag{9}$$

and, based on Amazon EC2 pricing,

$$p_{\text{bc IN}} = 0\,\$/\text{GB}, \tag{10}$$

$$p_{\text{bc OUT}} = 0.12\,\$/\text{GB}. \tag{11}$$

Since in the case of the SMSC the incoming traffic is approximately equal to the outgoing,

$$p_{\text{bc}} = (p_{\text{bc IN}} + p_{\text{bc OUT}})/2 = 0.06\,\$/\text{GB}. \tag{12}$$

From the above, it follows that

$$u_{\text{c}} = p_{\text{cc}}/p_{\text{co}} = 0.38/0.32 = 1.2; \tag{13}$$

$$u_{\text{b}} = p_{\text{bc}}/p_{\text{bo}} = 0.06/0.04 = 1.5. \tag{14}$$

Since the pricing of the computing capacity is charged on an hourly basis, we shall estimate a value for k that reflects the volume of transferred data (in GB) per unit of computing capacity over a period of an hour too, i.e. $T_{\text{CPU}} = 3600$ sec. Furthermore, while processing an SMS request, $V_{\text{ext}} = 1160$ B of data is transferred between the SMSC and the receiver, and $V_{\text{int}} = 1842$ B is transferred between the sender and the SMSC. Thus,

$$k = \lambda_{\text{inst}} T_{\text{CPU}} V_{\text{ext}} = 100 \times 3600 \times \frac{1160}{1\,073\,741\,824} = 0.39\,\text{GB per instance-hour, and} \tag{15}$$

$$\rho = \frac{V_{\text{int}}}{V_{\text{ext}}} = \frac{1842}{1160} = 1.59. \tag{16}$$

Now, having estimated the pricing parameters, let us go on to identify the cost-efficient division of the SMS demand. From eq. (6) we get:

$$\tau_{\text{c}}(q_{\min}) = \frac{(0.32 + 0.04 \times 0.39) \times 24 \times 30}{1.2 \times 0.32 + 0.04 \times 0.39(1.5 + 1.59 \times 2.5)} = \frac{241.6}{0.47} = 515. \tag{17}$$

Thus, in order to achieve a cost-optimal division of SMS demand out of $24 \times 30 = 720$ hours in month, the public cloud instances should be used during the "busiest" 515 hours (i.e. the hours with the highest load). These public instances are employed to serve the load exceeding the threshold value $\lambda_{\text{T}} = q n_{\max} \lambda_{\text{inst}}$, whereas during the remaining $(720 - 515 = 205)$ hours the private cloud capacity alone is sufficient to handle the load.

Using eq. (3) and an assumption that the maximum message rate is 500 messages/second, the total number of instances is estimated as $n_{\max} = \lceil 500/100 \rceil = 5$. Furthermore, based on the load distribution rearranged to be monotonically non-decreasing (see Figure 6), we get the λ_{T} corresponding to 205 hours:

$$\lambda_{\text{T}} = 190\,\text{requests per second},$$

from which it follows that

$$n_{\text{pr}} = q_{\min} n_{\max} = \lambda_{\text{T}}/\lambda_{\text{inst}} = 190/100 \approx 2, \tag{18}$$

and hence

$$q_{\min} = n_{\text{pr}}/n_{\max} = 2/5 = 0.4. \tag{19}$$

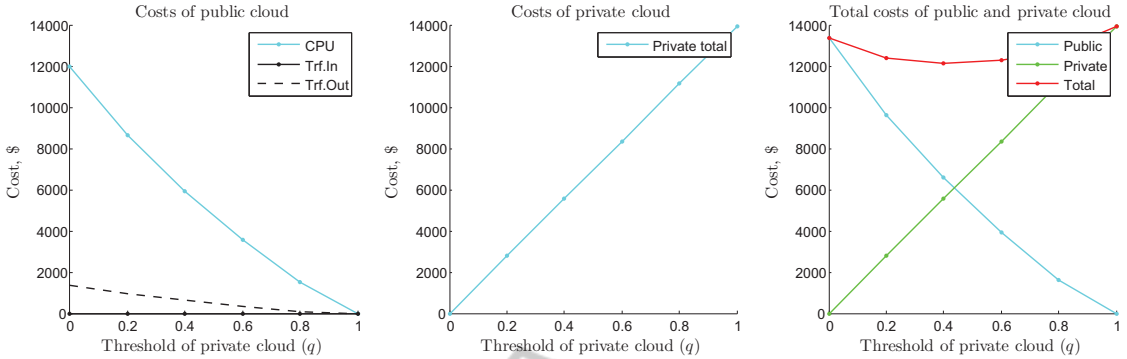

Figure 7: The yearly costs of the public portion (left) and the private portion (middle) of the hybrid cloud, as well as the overall hybrid cloud costs (right) plotted as a function of the private cloud threshold q.

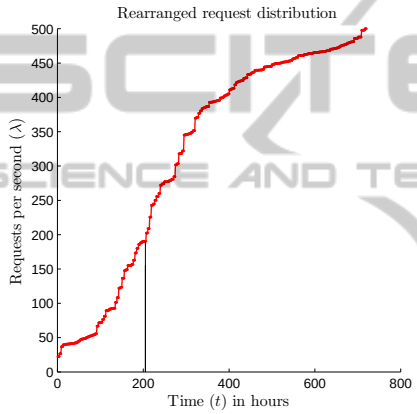

Figure 6: SMS request distribution, rearranged to be monotonically non-decreasing.

Thus, the cost-optimal private portion of the hybrid cloud is $q_{\min} = 0.4$, i.e. $n_{\text{pr}} = 2$ private computing instances should be acquired or leased for the purpose of SMSC. Whenever the SMS load is greater than the private cloud can efficiently handle, an instance from the public cloud needs to be launched.

This cost-optimal SMS load division is also visible in Figure 7, where the costs of the private and the public portions of the hybrid cloud, as well as the overall hybrid cloud costs are shown for different values of q. Note that the costs in Figure 7 are shown for a period of one year. The fully public cloud ($q = 0$) and the fully private cloud ($q = 1$) incur the costs of \$13 396 and \$13 980 per year, respectively, whereas the yearly cost of the cost-optimal load division is \$12 190. As can be seen, the overall hybrid cloud costs C is a (not strictly) convex function, and its minimum occurs when $q_{\min} = 0.4$, thus matching the results of the above estimations.

5 RESULT ANALYSIS

According to Section 4 the cost optimal hybrid cloud should have a composition where 40% of all traffic should be executed by a private cloud and 60% by a public cloud. Using our previous example, we note that originally we should have 2 private instances, but we must be prepared to update the system with 3 public cloud instances to cover the maximum load of 5 VMs or 500 requests per second. Table 2 presents number of required VMs for each load range defined by the *map* function presented in Algorithm 1. In our case the functionality of the *map* module is very simple and there is no need for a complex analysis as we should do in a general case (Mishra and Sahoo, 2011).

Table 2: VM capacity boundary values.

Workload (Req/s)	Number of VMs	Type
$[0, 200)$	2	private
$[200, 300)$	3	hybrid
$[300, 400)$	4	hybrid
$[400, 500)$	5	hybrid

To evaluate the results from the performance point of view, the dynamic resource provisioning models were applied to test the average message processing time during one day. Figure 8 presents the review results using both the reactive and proactive models. As shown in Figure 3 and Table 2, new resource allocations are needed three times per day when the example numbers are applied. Due to the startup time of a new public cloud instance, the reactive model suffers from an increased processing delay always when a new public cloud instance is launched, while the proactive model offers a stable performance. The results slightly improve when the number of instances increases because at the same time the average load per VM decreases. A lower load in VM results in a

better performance which fact is verified by Figure 4.

Our approach concentrates in the direct computation costs, and ignores several other factors (Martens and Teuteberg, 2011). In addition, the research is only dedicated to a black box approach and does not look at the dependencies between the sub modules (Mao and Humphrey, 2011; Van den Bossche et al., 2010). On the other hand, the presented algorithm is novel, and unlike the previous research (Chang et al., 2010; Yazir et al., 2010), the special requirements of the hybrid cloud in the telecommunication context are highlighted in the algorithm.

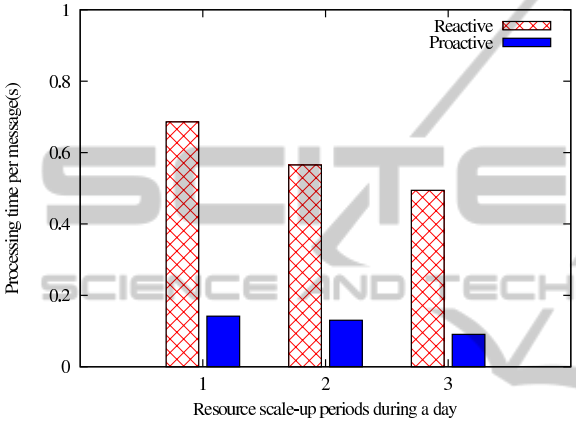

Figure 8: Review of hybrid cloud using reactive and proactive models.

6 DISCUSSION

The text messaging service was selected as the use case for good reasons. SMS is a widely spread service, and it can be well modeled and simulated. However, the SMS use case is not the most illustrative from the viewpoint of the hybrid cloud. This is due to the fact that the messaging traffic is periodic and the absolute differences in volume between minimum, base and peak loads are not huge. On the other hand, extreme peak workloads are very rare. In addition, the actual message size is moderate, accumulating a very small transmission load. The computation power required to manage an average SMS market segment is not large either. The advantages of the hybrid cloud will not be clearly visible in case of low traffic volume or variation. The message size as such is not very critical parameter (Hill and Humphrey, 2009).

The results can be applied to other services as well. In the mobile sector, the Multimedia Messaging Service (MMS) would be a logical extension to the use case. Unfortunately the MMS is still in the early market phase, and it is not yet well supported by the open source software community, creating challenges for the implementation of proof of concepts. Other interesting application examples are the global video on demand (VoD) (Li et al., 2011) and videoconference services that also require transcoding features (Cerviño et al., 2011). Volumes vary a lot in video service traffic (Li et al., 2011), and transmission load is significant. These facts enable a hybrid cloud architecture that can be geographically distributed.

The cost analysis has certain limitations. First of all, the cost analysis of the Amazon EC2 did not include prices of the reserved instances or the spot prices. Secondly, the model does not take into account the message retransmissions between MSCs and SMSC. According to the measurements (Zerfos et al., 2006), about 5% of the sent messages will never reach the target, and one or more retransmissions are needed in 22% of the cases. However, the performance of the latest SMSC systems can be expected to be a lot higher. Recently an operator reported a peak capacity of 19 000 messages per second[11] was delivered without congestion.

Furthermore, the costs of the load balancer have not been taken into account since these costs are constant. The cost of data transfer in the private portion of the hybrid cloud was assumed to be constant, too, independently of the occupied bandwidth, whereas a bandwidth-based charging was assumed in (Mazhelis and Tyrväinen, 2011). This may make the estimate of cost-optimal load division somewhat incorrect; however, since the data transfer costs in the private cloud are relatively low, also the inaccuracy of the estimation can be ignored.

The SMS arrival rate is assumed to be monitored on a per-minute basis, and the public cloud instances are launched or released depending on whether the arrival rate exceeds a certain threshold. However, public cloud computing instances are charged on an hourly basis, and hence, even if the period of high load - when the public instance is needed - lasts for a couple of minutes only, the whole hour will be charged. As a result, the evaluation of the cost-optimal load division somewhat underestimated the cost of public cloud instances; however, as the SMS distribution function rarely has sharp peaks, the underestimation can also be ignored.

Telecommunication operators can seek further reductions in costs by introducing a private cloud that can be shared between several operators. It can be also foreseen that infrastructure vendors will start to offer public, telecom specific clouds to their customers. The telecom clouds can be optimized for real time applications and their SLA requirements. Green

[11] http://www.itu.int/ITU-D/ict/newslog/CategoryView,category,SMS.aspx

cloud computing is another interesting research path that should be analyzed in the hybrid context (Mazzucco et al., 2010). These research ideas are left for future studies.

Finally, the usage of dynamic resource provisioning models in cloud computing deserves more research. The algorithms were not in the focus of this paper, but the predictive ARMA model was chosen to test the feasibility of our hybrid cloud solution. In the SMSC use case the performance of the ARMA algorithm was outstanding with an average error of 4.5%, but it remains to be seen how well it would operate with a random load that also includes a varying number of traffic peaks.

7 CONCLUSIONS

We have presented a novel Short Message Service Center (SMSC) architecture that is implemented in a hybrid cloud using Amazon EC2 as a public and OpenNebula as a private cloud. The hybrid cloud enables an efficient usage of computation resources depending on the load curve. The research is supported with an experimental setup that is implemented using an SMSC simulator. The proof of concept is complemented with a dynamic resource provisioning system that can utilize either reactive or proactive algorithm. The measurement data, gathered from the proof of concept, is further applied to verify the key parameters needed for the cost analysis section. Finally, the cost analysis proves that a hybrid cloud is a feasible alternative to support services where the traffic peaks are frequent.

However, due to the traffic characteristics of the SMS service, little benefit can be obtained from the hybrid approach in this particular use case. The SMS traffic varies relatively little, and furthermore, the message complexity and volume are on a minor scale. On the other hand, the SMS was only used here as a test case, and better results can be achieved with more dynamic traffic types, such as multimedia and video. In addition, the experimental system proves that a dynamic resource controller that uses prediction techniques, can significantly improve the system performance and in turn lower the costs. The research indicates that the dynamic hybrid cloud is an interesting alternative to static, either private or public, clouds. The optimal architecture can ultimately be decided after a careful analysis of the traffic pattern.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their valuable comments. The work is supported by Tekes (the Finnish Funding Agency for Technology and Innovation, www.tekes.fi) as a part of the Cloud Software Program (www.cloudsoftwareprogram.org) of Tivit (Strategic Centre for Science, Technology and Innovation in the Field of ICT, www.tivit.fi).

REFERENCES

Ardagna, D., Casolari, S., and Panicucci, B. (2011). Flexible distributed capacity allocation and load redirect algorithms for cloud systems. In *Proc. 4th International Conference on Cloud Computing*, Cloud 2011, pages 163–170. IEEE.

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., and Zaharia, M. (2010). A view of cloud computing. *Communications of the ACM*, 53(4):50–58.

Box, G. E. P. and Jenkins, G. (1990). *Time Series Analysis, Forecasting and Control*. Holden-Day, Incorporated.

Cerviño, J., Escribano, F., Rodríguez, P., Trajkovska, I., and Salvachúa, J. (2011). Videoconference capacity leasing on hybrid clouds. In *Proc. 4th International Conference on Cloud Computing*, Cloud 2011, pages 340–347. IEEE.

Chang, F., Ren, J., and Viswanathan, R. (2010). Optimal resource allocation in clouds. In *Proc. 3rd International Conference on Cloud Computing*, Cloud 2010, pages 418–425. IEEE.

Gabrielsson, J., Hubertsson, O., Más, I., and Skog, R. (2010). Cloud computing in telecommunications. *Ericsson Review*, 1:29–33.

Gong, Z., Gu, X., and Wilkes, J. (2010). PRESS: PRedictive Elastic ReSource Scaling for cloud systems. In *Proc. 6th International Conference on Network and Service Management*, CNSM 2010, pages 9–16. IEEE.

Hajjat, M., Sun, X., Sung, Y.-W. E., Maltz, D., Rao, S., Sripanidkulchai, K., and Tawarmalani, M. (2010). Cloudward bound: planning for beneficial migration of enterprise applications to the cloud. In *Proc. ACM SIGCOMM 2010*, SIGCOMM '10, pages 243–254. ACM.

Hamilton, J. (2010). Cloud computing economies of scale. Keynote at AWS Genomics & Cloud Computing Workshop, Seattle, WA, 08.06.2010, available from http://www.mvdirona.com/jrh/TalksAndPapers/JamesHamilton_GenomicsCloud20100608.pdf.

Hill, Z. and Humphrey, M. (2009). A quantitative analysis of high performance computing with Amazon's EC2 infrastructure: The death of the local cluster? In *Proc. 2009 10th IEEE/ACM International Conference on Grid Computing*, GRID, pages 26–33. IEEE.

Kaufman, L. M. (2009). Data security in the world of cloud computing. *IEEE Security and Privacy*, 7:61–64.

Khajeh-Hosseini, A., Greenwood, D., Smith, J. W., and Sommerville, I. (2010). The cloud adoption toolkit: Supporting cloud adoption decisions in the enterprise. *CoRR*, abs/1008.1900.

Kikuchi, S. and Matsumoto, Y. (2011). Performance modeling of concurrent live migration operations in cloud computing systems using PRISM probabilistic model checker. In *Proc. 4th International Conference on Cloud Computing*, Cloud 2011, pages 49–56. IEEE.

Li, H., Zhong, L., Liu, J., Li, B., and Xu, K. (2011). Cost-effective partial migration of VoD services to content clouds. In *Proc. 4th International Conference on Cloud Computing*, Cloud 2011, pages 203–210. IEEE.

Mao, M. and Humphrey, M. (2011). Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In *Proc. 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '11, pages 49:1–49:12. ACM.

Martens, B. and Teuteberg, F. (2011). Decision-making in cloud computing environments: A cost and risk based approach. *Information Systems Frontiers*, pages 1–23.

Mazhelis, O. and Tyrväinen, P. (2011). Role of data communications in hybrid cloud costs. In *Proc. 37th EUROMICRO Conference on Software Engineering and Advanced Applications*, pages 138–145. IEEE.

Mazzucco, M., Dyachuk, D., and Deters, R. (2010). Maximizing cloud providers' revenues via energy aware allocation policies. In *Proc. 3rd International Conference on Cloud Computing*, Cloud 2010, pages 131–138. IEEE.

Mishra, M. and Sahoo, A. (2011). On theory of VM placement: Anomalies in existing methodologies and their mitigation using a novel vector based approach. In *Proc. 4th International Conference on Cloud Computing*, Cloud 2011, pages 275–282. IEEE.

Moreno-Vozmediano, R., Montero, R. S., and Llorente, I. M. (2009). Elastic management of cluster-based services in the cloud. In *Proc. 1st workshop on Automated control for datacenters and clouds*, ACDC '09, pages 19–24. ACM.

Mouly, M. and Pautet, M.-B. (1992). *The GSM System for Mobile Communications*. Telecom Publishing.

Murphy, M. (2010). Platform-as-a-Service for Telcos. Keynote at Cloud Asia, Singapore, 05.05.2010, available from http://cloudasia.ngp.org.sg/2010/programs_track2.php.

Pacheco-Sanchez, S., Casale, G., Scotney, B. W., McClean, S. I., Parr, G. P., and Dawson, S. (2011). Markovian workload characterization for QoS prediction in the cloud. In *Proc. 4th International Conference on Cloud Computing*, Cloud 2011, pages 147–154. IEEE.

Paivarinta, R. and Raivio, Y. (2011). Performance evaluation of NoSQL cloud database in a telecom environment. In *Proc. 1st International Conference on Cloud Computing and Services Science*, Closer 2011, pages 333–342. SciTePress.

Raivio, Y. and Dave, R. (2011). Cloud computing in mobile networks - case MVNO. In *Proc. 15th International Conference on Intelligence*, ICIN 2011, pages 253–258. IEEE.

Roy, N., Dubey, A., and Gokhale, A. (2011). Efficient autoscaling in the cloud using predictive models for workload forecasting. In *Proc. 4th IEEE International Conference on Cloud Computing*, Cloud 2011. IEEE.

Strebel, J. and Stage, A. (2010). An economic decision model for business software application deployment on hybrid cloud environments. In Schumann, M., Kolbe, L. M., Breitner, M. H., and Frerichs, A., editors, *Multikonferenz Wirtschaftsinformatik 2010*, pages 195–206. Universitätsverlag Göttingen.

Van den Bossche, R., Vanmechelen, K., and Broeckhove, J. (2010). Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads. In *Proc. 3rd International Conference on Cloud Computing*, Cloud 2010, pages 228–235. IEEE.

Venugopal, S., Li, H., and Ray, P. (2011). Auto-scaling emergency call centres using cloud resources to handle disasters. In *Proc. 19th International Workshop on Quality of Service*, IWQoS '11, pages 34:1–34:9. IEEE Press.

Walker, E. (2009). The real cost of a CPU hour. *Computer*, 42(4):35–41.

Weinman, J. (2011). Mathematical proof of the inevitability of cloud computing. Online report, 08.01.2011, available from http://www.joeweinman.com/Resources/Joe_Weinman_Inevitability_Of_Cloud.pdf.

Yazir, Y. O., Matthews, C., Farahbod, R., Neville, S., Guitouni, A., Ganti, S., and Coady, Y. (2010). Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis. In *Proc. 3rd International Conference on Cloud Computing*, Cloud 2010, pages 91–98. IEEE.

Zerfos, P., Meng, X., Wong, S. H., Samanta, V., and Lu, S. (2006). A study of the short message service of a nationwide cellular network. In *Proc. 6th ACM SIGCOMM conference on Internet measurement*, IMC '06, pages 263–268. ACM.

Zhang, H., Jiang, G., Yoshihira, K., Chen, H., and Saxena, A. (2009). Intelligent workload factoring for a hybrid cloud computing model. In *Proc. 2009 Congress on Services - I*, pages 701–708. IEEE.