Telco Clouds: Modelling and Simulation

Jakub Krzywda[1], William Tärneberg[2], Per-Olov Östberg[1], Maria Kihl[2], and Erik Elmroth[1]
[1]*Dept. of Computing Science, Umeå University, SE-901 87 Umeå Sweden*
[2]*Dept. of Electrical and Information Technology, Lund University, SE-223 63 Lund, Sweden*
{*jakub, p-o, elmroth*}*@cs.umu.se,* {*william.tarneberg, maria.kihl*}*@eit.lth.se*

Keywords: Mobile cloud computing, Telecommunication infrastructure, Cloud infrastructure, Modelling, Simulation

Abstract: In this paper, we propose a telco cloud meta-model that can be used to simulate different infrastructure configurations and explore their consequences on the system performance and costs. To achieve this, we analyse current telecommunication and data centre infrastructure paradigms, describe the architecture of the telco cloud and detail the benefits of merging both infrastructures in a unified system. Next, we detail the dynamics of the telco cloud and identify the components that are the most relevant from the perspective of modelling performance and cost. A number of well established simulation technologies exist for most of the telco cloud components, we thus proceed with surveying existing models in an attempt to construct a suitable composite meta-model. Finally, we present a showcase scenario to demonstrate the scope of the telco cloud simulator that we have implemented.

1 INTRODUCTION

Recent technological developments have enabled a union of telecommunication and cloud computing. Joint management of telecommunication infrastructure, such as Radio Base Stations (RBS), and Data Centres (DC) may help to achieve better performance of hosted applications and reduce the operation costs. Depending on the perspective and focus of the viewer, this paradigm is known under different names, e.g., ubiquitous cloud computing (Li et al., 2013), fog computing (Bonomi et al., 2012), mobile edge computing (Drolia et al., 2013), mobile cloud computing (Dinh et al., 2013), or, the name used in this paper: telco cloud computing (Bosch et al., 2011).

Despite the interest in this paradigm, there are no simulation models capable of simultaneously modelling the dynamics of Mobile Devices (MD), placement and capacity of DCs, and network infrastructure. Understanding of these relations is important for telco cloud stakeholders, e.g., Infrastructure Providers (IP) can use that knowledge to reduce infrastructure costs, while still delivering competitive performance.

We propose a meta-model of the telco cloud which facilitates experimentation and evaluation of possible configurations, such as placement and capacity of DCs in a joint telecommunication-cloud infrastructure. The meta-model uses existing, well established simulation models, e.g., for Radio Access Networks (RANs) or DCs, for modelling of the aforementioned individual parts of the infrastructure behaviour.

The contributions of this paper are: describing the dynamics of the telco cloud, including Quality of Service (QoS) and the associated costs of this paradigm (Section 4); surveying existing models of telco cloud building blocks (Section 5); and establishing a meta-model that captures the described dynamics using existing and composite models (Section 6).

This paper is structured as follows. Section 2 outlines the architecture of the proposed telco cloud. Next, the simulation motivations, challenges, and requirements that the meta-model has to fulfill are presented in Section 3. Section 4 describes the telco cloud dynamics, followed by a survey of existing models in Section 5. Section 6 introduces the telco cloud meta-model. Section 7 presents a showcase simulation scenario, using a prototype implementation of the introduced meta-model. In Section 8 we list a few relevant research topics that can be explored using the proposed model and conclude the paper.

2 ARCHITECTURE OF THE TELCO CLOUD

In this section we present issues with current telecommunication and cloud infrastructures, an overview of

the proposed telco cloud topology, and how the telco cloud paradigm can help to remedy these issues.

2.1 Current Infrastructure

Currently, telecommunication and cloud computing infrastructures are separated and managed independently. The telecommunication infrastructure is placed in close proximity to end users and is built using specialised hardware. The cloud computing infrastructure consists of remote DCs that are significantly geographically separated from end users, consists of commodity hardware, and is connected with the telecommunication infrastructure via the Internet.

We identify several issues of the current infrastructure. Performance (especially latency) of cloud services is not predictable, which makes computation offloading difficult. All data processed in the cloud need to be sent over the Internet to DCs, which add communication latency. For the Internet of Things, with millions of sensors generating huge amounts of data, the volume of traffic can cause network congestion. Specialised telecommunication hardware is expensive and hard to upgrade.

As a consequence of the performance bottlenecks, particularly latency sensitive applications such as industrial process control and Augmented Reality context recognition applications have mostly not yet been cloudified. The low latency, jitter free, and high throughput connections required by such applications cannot be provided by the existing telecommunication and cloud infrastructures (Barker and Shenoy, 2010). Moreover, compute and battery resources in MDs are limited and coupled. An approach to augmenting MD's capabilities is to offload the execution of applications to a cloud infrastructure. Such performance augmentation can only be seamless if the incurred communication latency is low enough and service availability is high.

Devices are at an increasing rate gaining access to the Internet (Atzori et al., 2010). Anything from small sensors to petrol pumps, is being connected to the Internet as a mean to communicate and monitor its quantified individual performance metrics. The gathered data is typically used to assess the performance of the system(s) they are a part of and/or the user they belong to. Most of the connected devices are generating correlated contextual information, incurring large amounts of Wide Area Network's (WAN) traffic. The traffic often converges to a handful of DCs for analysis and processing which introduces congestion in the capacity-sparse and increasingly congested RANs, core networks, and WANs. Additionally, a proportion of the transported information is only locally relevant and will not have any or very little entropy to the service in the remote DC.

Wireless access network virtualisation and cloudification of Telecom equipment and services proposed by (Wang et al., 2013), requires careful placement of compute capacity as not to introduce significant propagation delay. The placement of the compute nodes has to reflect the demand for Telecom- and cloud-services in the geographic area which the RAN covers. Current Telecom signalling standards and remote DCs do not interoperate well and are often not able to meet Telecom latency requirements.

2.2 Introducing the Telco Cloud

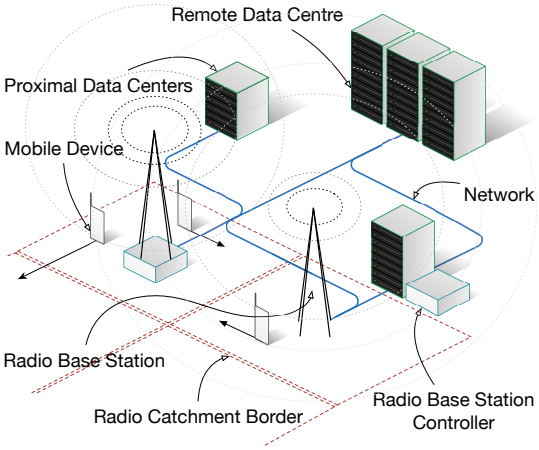


Figure 1: Overview of telco cloud.

A telco cloud is an infrastructure consisting of MDs, stationary devices (e.g. sensors generating data), access networks, intermediate WANs (connecting the access networks to the backbone networks), backbone (Internet) networks, and DCs. We here include two main types of DCs: Remote Data Centres (large DCs located far from the access networks) and Proximal Data Centres (smaller DCs located close to the access networks).

The telco cloud (Patel et al., 2014; Drolia et al., 2013) topology paradigm proposes a closer integration between access networks and a cloud infrastructure than the current topological model where the Telecom and cloud infrastructures are unaware of each other. With the coexistence of the cloud infrastructure, telecommunication functionality can be virtualised and augmented to the adjacent DCs. When virtualising RAN functions large portions of an RBS and the RAN control functions can be executed in a DC (Baroncelli et al., 2010). Essentially hosting the RBS Controller in a generic Infrastructure as a Service (IaaS) Cloud infrastructure.

Cloud capacity will reside in geographically distributed DCs. The telco cloud topology is composed of multiple DCs that are dispersed in a mesh-structure, ranging from complete adjacency with the Telecom infrastructure, Proximal DCs, to more traditional, Remote DCs, as depicted in Figure 1.

A group of telco cloud DCs can either be provisioned and load balanced as one resource or act as independent DCs. In the former case, a control plane will need to coordinate services and the shared resources by for example optimising locality by geographically placing services to end users. Proximal DCs will have less capacity than the traditional remote DCs, and will have to be managed in a distributed and coordinated manner.

The cloud services and Proximal DCs are accessed through the RAN and cater for the MDs within it. The MDs are assumed to possess varying degrees of mobility, with an equivalent likelihood of passing between a particular set of macro/micro-RBS[1] over time. In an effort to be able to enforce DC management constraints and to globally avoid unnecessary migration, an MD is not strictly associated with the closest DC or the DC that its current RBS cell is associated with.

The telco cloud infrastructure topology will need to reflect the capacity and latency objectives of the virtualised RBSs, and to give optimal access to the telco cloud-hosted services given the networks local capability and diversity. The prevailing 4G/LTE topology is centred around hierarchical macro- and micro-cells which very much resembles that of its predecessors. The telco cloud topology will evolve with mobile access technology generations, shifting and/or dispersing compute capacity at various levels of mobile, Metropolitan Area Network (MAN), and WAN networks to best suit the prevailing services and throughput channels.

2.3 Benefits of the Telco Cloud

We identified four main benefits of the telco cloud. First, it provides cloud applications with better and more predictable performance. Second, it supports computation off-loading for resource-bounded and low-power mobile devices. Third, the telco cloud reduces network utilization by processing part of data closer to its producer or consumer. Fourth, it enhances cost-efficiency and flexibility of telecommunication infrastructure.

Thanks to a geographically distributed cloud infrastructure with a progressive proximity to the end customer, application developers and Mobile Network Operators (MNO) can take advantage of the significantly lower round-trip times. Moreover, the average network throughput will increase when communicating with Proximal DCs in comparison to traditional remote DCs. Additionally, if a user seeks to offload MD applications they will benefit from a more responsive low latency communication link to where the code is executing.

The large amount of information independently generated by connected sensors destined for remote DCs can be filtered through the intermediate hierarchical cascade of DCs to prevent congestion in the intermediate WAN. Data that is only locally relevant can be kept and consumed locally, while redundant and highly covariant information can be more easily identified in a local context and discarded.

Through the telco cloud traditional proprietary hardware-bound Telecom services can be virtualised, migrated, and executed in adjacent Proximal DCs. Multiple RBSs can be consolidated to increase the aggregate utilisation of the infrastructure. Executing RBSs on cloud infrastructure will allow for greater use of cost-effective commodity hardware and generic software. With the availability of the telco cloud, an RBS in future mobile infrastructure generations will only consist of a radio interface. The management of the RAN, individual radio channels, signalling, services, and signal processing, can be all virtualised and executed in a Proximal DC in the vicinity.

3 SIMULATION CHALLENGES

Simulation of a telco cloud is motivated by several factors, e.g., lack of existing infrastructure and appropriate control plane standards, as well as the economic implausibility of creating a large and configurable physical test bed. The desired simulator has to fulfil many requirements, such as, to be able to simulate hundreds of thousands of entities that have different types (e.g. MD, network, DC) at very fine grained time granularity (milliseconds) for long periods of time (hours) in parallel. These requirements cause simulation of telco clouds to be very challenging. We here motivate and describe identified requirements and challenges in simulation of the telco cloud. Addressing the challenges as well as fulfilling the requirements is crucial while designing a meta-model and implementing a simulator.

Telco cloud stakeholders will benefit from being able to investigate the consequences of possible infrastructure configurations. For example, IPs responsible for building and maintaining the infrastructure

[1] What is considered a traditional rural/urban cell, constituted by a high power RBS, mounted on a tower.

may use the simulator for planning placement and capacity for new DCs, as well as modifying capacity and connectivity of existing DCs. Service providers that use infrastructures to host services are interested in comparing different strategies for placement of application components. Moreover, application developers that develop mobile applications utilising a telco cloud, need to determine what components of a mobile application could benefit from offloading. To answer these and similar questions, tests with various infrastructure configurations need to be performed and results compared. There are two options for performing these tests: by simulation or by running them in real test beds.

Currently, there is no existing infrastructure that can be used for testing the telco cloud. Creating such a physical testbed for large-scale testing of a telco cloud in different configurations is economically infeasible and small-scale testbeds will not be able to capture phenomena occurring in reality, such as, user mobility patterns or latency issues.

For these reasons, we believe that simulation is the most feasible option to evaluate the telco cloud. However, we have identified several requirements that make simulation of telco clouds challenging. First of all, the scale of simulation is large in terms of number and types of entities. The simulation of telco cloud has to concurrently cover hundreds of thousands of MDs moving around a simulated area, each generating requests; hundreds of RBSs providing an access to the core network; and tens of DCs, running services that process requests. Another challenge is the ratio between time precision and length of simulation. We are interested in a very fine-grained latency simulation, that requires precision of at least milliseconds. However, to capture the daily patterns of MD movement (e.g., moving between home, work, and shops) caused by migrations of users carrying them, the whole system needs to simulate several run-time hours. Moreover, a simulation of application statefulness, and of transferring those states when MDs are moving, have not yet been described in the literature and requires new models to be developed.

4 DYNAMICS OF THE TELCO CLOUD

Before constructing a meta-model of the telco cloud we first need to understand telco cloud fundamental dynamics, construed as the relations between system's input, configuration and output. Later, we will use the knowledge of these dynamics to build the intended simulation meta-model.

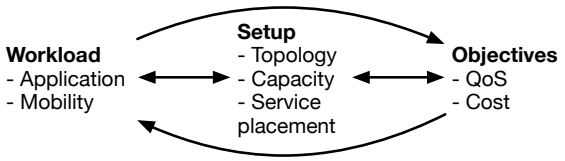


Figure 2: Dependencies between elements of a telco cloud.

Figure 2 visualizes the dynamics inside the telco cloud. The *workload* is an input to the system that IPs do not have influence over. It includes: applications, with a request generation model (rate and size), a resource requirements model (the amount of resources needed to process a request), and application statefulness (the overhead of transferring user's state between DC); as well as, the mobility of users carrying MDs.

Next, *objectives* describe required output characteristics of the system. We have identified two fundamental objectives. Firstly, QoS, which imposes performance requirements, e.g., latency or throughput through Service Level Objectives (SLO). Secondly, the monetary cost for IP associated with energy consumed for computation and maintenance of an infrastructure. The objectives can be used when constructing an optimisation problem with QoS as conditions and cost as the function that should be minimised.

Finally, *setup* is that part of the system that can be adjusted by designers or operators to achieve desired objectives when processing existing workloads. Setup includes topology, location, capacity of DCs and the network that interconnects MDs and RBSs with DCs, as well as resource management policies that control placement and migration of services.

The above mentioned elements are all dependent on each other. First, the setup of a telco cloud is related to the existing workload. The capacity of a DC is defined by the resource demands of the services, e.g. how memory-, CPU-, network-, and disk-intensive the services are. The locations and topology of DCs are defined by the geographic and demographic scope of the services, the number of MDs that reside in that domain, and the capacity of the associated telecommunication infrastructure.

Secondly, workload influences objectives. E.g., user mobility is inducing delay during service migration and potentially causes QoS penalties. Moreover, application statefulness introduces additional costs of storing data in a DC and transmitting data between Proximal DCs. It may also increase latency due to an additional time necessary to send the state data before processing of the request can begin.

Finally, objectives depend on the setup: QoS is proportional to the proximity and capacity of DC – a smaller DC catchment (the geographic area the DC serves) translates to greater locality and reduced prop-

agation latency, while higher capacity allows hosting of more services. Moreover, the capacity and catchment of DCs determine telco cloud costs. Costs are proportional to dispersion of computing capacity. Firstly, there is an overhead of each DC, irregardless of its capacity, e.g., building, cooling infrastructure, and connection to energy or network. Secondly, dispersion increases costs of maintenance, e.g., technicians have to travel between locations. Therefore, costs are proportionally higher in smaller, dispersed DCs because of high initial costs and proportionally lower in huge, centralized DCs due to the economy of scale (Armbrust et al., 2010).

5 EXISTING MODELS AND SIMULATORS

To facilitate creation of a simulation meta-model that incorporates workload, setup, and objectives of the telco cloud described in the previous section, we here survey existing models in the following categories: application request generation and resource requirements, MD mobility, networks, DCs, and costs of infrastructure.

Most of the models and simulators are assigned to only one of the above mentioned categories, however capabilities of four surveyed simulators extend to many categories, so we summarise them in Table 1.

Table 1: Overview of surveyed simulators.

Framework	RG	RR	M	N	DC
NS-3	✓		✓	✓	
OMNeT++	✓		✓	✓	
CloudSim		✓			✓
GreenCloud		✓			✓

RG – Request Generation, RR – Resource Requirements, M – Mobility, N – Network, DC – Data Centre.

5.1 Workload Models

Applications running in the telco cloud consist of a mobile client and a server processing offloaded computations that have to be modelled from two perspectives: *request generation* that describes how requests are generated by end users and sent over the network to the DCs; and *resource requirements* that describes how much computational resources are needed to process the requests.

Request Generation

There are several application behavioural models and traffic models for applications ranging from YouTube (Abhari and Soraya, 2010) to web browsing (Lee and Gupta, 2007). The models capture the user's behaviour by primarily representing interaction times or the timing clicks through a stochastic process, often Poissonian in nature. A user behavioural model can be further refined by introducing a stochastic model for how long time a user consumes a certain type of content. Additionally, the transition between types of content is often modelled as a Markov process.

Furthermore, the traffic characteristics are commonly modelled with multiple stochastic processes, encompassing the number of packets in a session, and the size of each packet. Traffic models are often referred to as either closed or open looped. In an open loop model, the generation of each new session is typically a Poission process independent of the resulting DC action. Conversely, in a closed loop model, the generation of new sessions is dependent on timing of the response from the DC and thus the properties of the previously generated session.

In NS-3 (Riley and Henderson, 2010) and OMNeT++ (Varga et al., 2001), which are packet-level event driven network simulators, a node can act as either a client or server, by the mechanism of either sending packets provided by a stochastic model, at a given rate, within a certain time period, and at a certain interval, or processing received packets from a buffer, at a given rate. Both server and client models can be augmented with a more complex system of queues to such an extent that they can represent an abstract DC that hosts multiple applications.

Resource Requirements

CloudSim (Calheiros et al., 2011), which is a simulator of cloud infrastructure, provides an application model that describes computational requirements – the amount of resources that needs to be available (e.g. number of cores, memory and storage); and communicational requirements – the amount of data that needs to be transferred. GreenCloud (Kliazovich et al., 2012), which is a packet level simulator based on NS-2, apart from computational and communicational requirements, describes also QoS requirements, expressed by an execution deadline. The application model may also include the size of the code that has to be offloaded and dependencies on other services, e.g., in terms of amount of data that has to be sent or received (Kovachev, 2012).

Mobility

The NS-3 and OMNeT++ nodes described above can be set into motion given a certain stochastic mobility model. They can for example traverse the space

as pedestrians, or auto mobiles, with corresponding velocity and rate of change. The spatial relationship between nodes and RBS affects the prevailing channel properties and RBS-to-node associations. Node mobility will also result in handover between RBSs, which in turn will alter the paths of the node-generated workload in the network.

5.2 Setup Models

Below, we describe existing models and simulators of networks and DCs, which can be used to configure the setup of the telco cloud meta-model.

Network

There are several well-established event-driven frameworks that are capable of modelling computer networks, mobile networks, applications, packet-level network traffic, infrastructure, and independent users. The two primary examples are, mentioned in the previous section, NS-3 and OMNeT++. Although with some doubt regarding their accuracy (Weingartner et al., 2009), these two are commonly deployed in academic network research and provide detailed results on network utilisation, throughput, congestion, and latency.

Both NS-3 and OMNeT++ are comprehensive packet-level network simulation frameworks that include wired and wireless standards, and are able to simulate communication channel conditions. Furthermore, both frameworks have detailed models for channel definition, such as propagation delay, interference, data rate, and medium access schemes. In addition, to a varying degree, NS-3 and OMNeT++, by default or through extension, support control plane signalling for a number of wireless standards and complex network topologies.

Both frameworks have support for modelling different types of network nodes, ranging from computers to routers and switches. Each edge and node pair has a defined communication and medium access standard, such as TCP/IP and Ethernet. Each packet that is sent over the network is treated in accordance with the prevailing network and transport protocols and routing standard. In both, the event of arrival and departure of packets drives the simulation clock.

Even though they are prevalent in academic network research they are cumbersome to manipulate and to adapt for new standards and topologies. Furthermore, they require detailed configuration of all communication modes as well as node behaviour, making it very time-consuming to implement and verify systems with different levels of abstractions.

The telco cloud topology is yet to be defined with unspecified control planes, it would thus be counter-intuitive and time consuming to implement telco cloud topologies in either NS-3 or OMNeT++. In some instances, some modules would have to be completely redesigned, and others would have to be specified to a much greater detail than the telco cloud can offer at this stage.

Data Centre

The purpose of this section is to survey the DC models that are the most suitable for inclusion in the telco cloud meta-model. An extensive list of mathematical models, simulation approaches, and test beds can be found in (Sakellari and Loukas, 2013), while (Ahmed and Sabyasachi, 2014) provides a survey of twelve cloud simulators. After careful examination, we chose the ones that best suit our goals.

We compare DC models and simulators based on descriptions provided by the authors of the simulators. For each model we describe: *Resource Provisioning* – what resources are included and how they are modelled; *QoS* – what performance indicators are measured; *Costs* of computation in the DC; *Performance* of simulator – an estimation of the time needed to perform a simulation.

CloudSim is an event-based simulator implemented in Java, for simulation of cloud computing system and application provisioning environments.
Resource Provisioning. The CloudSim simulation layer offers dedicated management interfaces for CPU, memory, storage and bandwidth allocation, as well as, defining policies in allocating hosts to Virtual Machines (VM) – VM provisioning. Hosts are described by processing capabilities (in MIPS) and a core provisioning policy, together with an amount of available memory and storage. A model supports time-sharing and space-sharing core provisioning policies on both host and VM levels.
Latency (QoS). The latency model is based on conceptual networking abstraction, where the communication delays between each pair of entity type (e.g. host, storage, end-user) are described in a latency matrix as a constant value expressed in simulation time units (e.g. milliseconds).
Costs. CloudSim provides a two-layered cost model, where the first layer relates to IaaS, with costs per unit of resources, while the second one relates to Software as a Service (SaaS), with costs per task units (application requests). This model allows calculation of the costs of using the cloud from the end-user perspective or the revenue from the IP perspective.
Performance. CloudSim is able to perform large-scale simulations, e.g., it can instantiate an experiment with

1 million hosts in 12 seconds. Moreover, memory usage grows linearly with the host number and even with 1 million hosts it does not exceed 320 MB.

CloudAnalyst (Wickremasinghe et al., 2010) is a simulator of geographically distributed large-scale cloud applications, developed with Java and that utilises CloudSim and SimJava.

Resource Provisioning. Cloud Analyst uses the same resource provisioning model as CloudSim.

Latency (QoS). A latency model allows configuration of network delays, available bandwidth between regions, and current traffic levels. CloudAnalyst facilitates experiments with latency by producing following statistical metrics: average, minimum, and maximum response time of all user requests; and response time grouped by time of the day, location, and DC.

Costs. CloudAnalyst supports calculation of costs for using cloud resources, such as cost per VM per hour and cost per Gigabit of data transfer.

Performance. To improve performance of simulation entities are grouped at three levels: clusters of users, cluster of requests generated by users, and clusters of requests processed by VM.

GreenCloud is a packet level simulator based on NS-2, for simulation of energy-aware clouds.

Resource Provisioning. Servers are modelled as a single core node with defined processing power limit (in MIPS or FLOPS), size of memory and storage, and implementing different task scheduling mechanisms.

Latency (QoS). Full support for the TCP/IP protocol reference model is provided and thanks to that the simulator is able to calculate communication latency with high accuracy.

Costs. GreenCloud allows detailed modelling of energy consumption by implementing energy models for every DC element.

Performance. Given that GreenCloud has to simulate the full stack of Internet protocols, each simulation only takes in the order of tens of minutes for a DC with a few thousands of nodes.

BigHouse (Meisner et al., 2012) is a discrete event simulator for high-abstraction-level simulation of DC systems, based on stochastic queuing simulation methodology.

Resource Provisioning. The simulator represents the DC infrastructure, e.g. servers and racks, using an object-oriented hierarchy. Various DC architectures are supported and there is also a possibility of extending them by modelling new functionality (e.g. power models). Model parameters, such as number of cores, can be supplied using configuration files.

Latency (QoS). BigHouse is capable of measuring response and waiting times (average value and 95$^{\text{th}}$percentile) for each simulated task.

Costs. Modelling of power management policies is supported and the simulator can estimate energy consumed by tasks.

Performance. Performance of BigHouse depends strongly on the desired accuracy and confidence of output metrics. For example, simulating a cluster with 10 000 servers takes between minutes and hours, depending on the desired level of confidence.

5.3 Costs Models

The above mentioned DC models focus mostly on the costs of running applications in DCs from the end-user perspective. Since we want to model costs of DCs from the IP point of view, additional models are needed for capital expenditures (CAPEX) and operating expenditures (OPEX).

CAPEX includes costs of infrastructure that needs to be built and servers that have to be bought.

Costs. Costs of building, power distribution (including UPS), and cooling can be estimated using a following equation: $\$200M \cdot (1+c_m)/a_i$, where c_m is the cost of money[2], and a_i is the time of infrastructure amortisation [in years] (Greenberg et al., 2008).

Server Costs. Costs of servers can be modelled as $n_s \cdot p_s \cdot (1+c_m)/a_s$, where n_s is the number of servers, p_s is the price of one server [in \$], c_m is the cost of money, and a_s is the time of server amortisation [in years] (Greenberg et al., 2008).

OPEX consists of power and personnel costs.

Power Costs. To estimate costs of power, the following equation can be used, $n_s \cdot pc_s/1000 \cdot PUE \cdot p_{KWH} \cdot 24 \cdot 365$, where n_s is the number of servers, pc_s is the power consumption of one server [in W], PUE is Power Usage Efficiency, and p_{KWH} is the price of electricity [in \$ per KWH] (Greenberg et al., 2008).

Personnel Costs. Costs of personnel can be calculated using $M_1 \cdot C_1 + M_2 \cdot C_2 + M_3 \cdot C_3$, where M_1 is the number of IT personnel per rack, M_2 is the number of facility personnel per rack, M_3 is the number of administrative personnel per rack, and C_1, C_2, C_3 are the average costs per person for each of the above mentioned categories (Patel and Shah, 2005).

6 TELCO CLOUD META-MODEL

We here detail how we have composed the above surveyed models into a telco cloud meta-model. Figure 3 depicts the visualisation of the meta-model. MDs, such as cell phones or laptops, are carried by end-users, who are in motion. The MDs generate requests

[2]Cost of money is the rate of interest or dividend payment on borrowed capital.

which are sent over the network to a DC. It is also possible that requests are generated by sensors that may be static (e.g. traffic cameras) or mobile (e.g. trains). The requests are processed in the DC and the response is sent back to the MD or sensor. Processing requests, in case of statefull applications, generates a user state, that has to be migrated with the end-user if he moves to another DC.

The primary objective of the meta-model is to capture the interactions between application workload, MD mobility, network topology, and DC characteristics, and their influence on QoS and costs of telco cloud. The parameters that define the meta-model are presented in Table 2 and described in detail below.

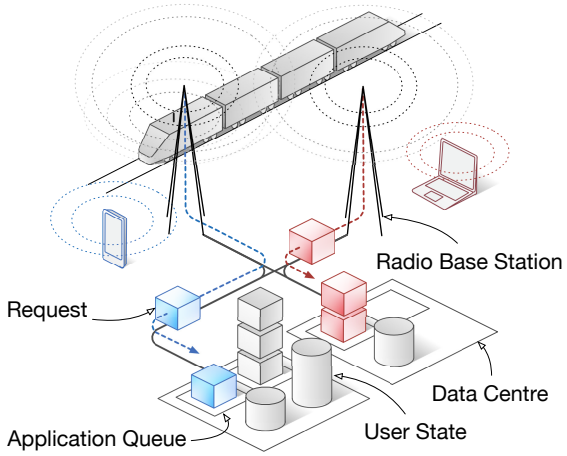


Figure 3: Visualisation of telco cloud meta-model.

6.1 Workload Model

The first group of parameters in Table 2 describes the mobility of end-users carrying MD and the characteristics of requests generated by these MDs.

Request Generation

Many services may run in the telco cloud at the same time and their number is defined by N_{ser}. We model a service application as a stateful web service. Each session is separated in time with a Poisson process λ_{ses} (Reyes-Lecuona et al., 1999). Each session produces N_{req} requests, sampled from an inverse Gaussian distribution, where each request is separated in time by Log-Normal distributed delay D_{req} in seconds. The size of each request is given by S_{req} KB and is drawn from a Pareto distribution.

Resource Requirements

To model application resource requirements we propose a linear model specifying the needed amount of resources, both for an idle service and per processing each request. An idle service uses CPU_{idle} CPU operations, mem_{idle} amount of memory, and $disk_{idle}$ amount of storage. Additionally for each processed request, the service uses CPU_{req} CPU operations, mem_{req} amount of memory, and $disk_{req}$ amount of storage. The amount of user's state data created by each request is defined by $state$ and expressed in absolute value or percentage of request size S_{req}.

Mobility

The network is populated by N_{MD} MDs, each subscribing to a subset of the N_{ser} available services. The 2-dimensional, multi modal, mobility model detailed in (Bettstetter, 2001) provides us with an on-average uniform distribution of users, with movement proportional to the duration of a session and the scale of the mobile network. The aforementioned model defines the properties of an MD's movement. A MD's momentary movement is defined by its velocity constituted by the current speed s and current direction θ. Changes in mobility are defined by multiple stochastic processes that describe the duration of its state. An entity's speed s is independent of direction θ and is maintained for T_s seconds, after which acceleration a is applied between a_{min} and a_{max} for time T_a, until it reaches s_{min} or s_{max}. Furthermore, direction θ is maintained for time T_θ until the next change-event where the direction θ is altered for T_ω seconds with at the rate of ω radians per second. T_s, T_a, T_θ, and T_ω, describing the timing of each change-event, are set for each mobility mode, and are each defined by a probability distribution bounded by a maxima and minima.

6.2 Setup Model

The second group of parameters in Table 2 characterises the network and DCs.

Network

In our model, the core network introduces a cumulative propagation, switching, and routing delay and it is modelled with a Weibull delay D_{net} in multiples of the number of network nodes between the source and the destination (Papagiannaki et al., 2003). The network distance between RBSs is equal to the cell dimension d_{RBS}. The associated RBSs are equidistant to their common DC, and are for the sake of simplicity assumed to be separated by one network edge.

Furthermore, forthcoming cell planing practices aim to increase area energy efficiency by favouring smaller cells in urban areas (Shahab et al., 2013; Fehske et al., 2009). Our model employs a small

Table 2: Fundamental meta-model parameters.

Type	Parameters	Unit	Description
WORKLOAD			
Request Generation	N_{ser}		Total number of services
	λ^i_{ses}, where $i = 1,2,\ldots,N_{ser}$	s	Session arrival rate to DC
	N^i_{req}, where $i = 1,2,\ldots,N_{ser}$		Number of requests per user session
	S^i_{req}, where $i = 1,2,\ldots,N_{ser}$	KB	Size of requests for a given service
	D^i_{req}, where $i = 1,2,\ldots,N_{ser}$	s	Inter-request time
Resource Requirements	CPU^i_{idle}, CPU^i_{req}, where $i = 1,2,\ldots,N_{ser}$	MI	Number of CPU cycles used by service
	mem^i_{idle}, mem^i_{req}, where $i = 1,2,\ldots,N_{ser}$	MB	Size of memory used by service
	$disk^i_{idle}$, $disk^i_{req}$, where $i = 1,2,\ldots,N_{ser}$	MB	Size of storage used by service
	$state^i$, where $i = 1,2,\ldots,N_{ser}$	MB	Size of user's state produced per request
Mobility	N_{MD}		Number of Mobile Devices
	s^i_t, a^i_t, θ^i_t, ω^i_t, where $i = 1,2,\ldots,N_{MD}$		Movements of Mobile Devices
SETUP			
Network	N_{RBS}		Number of Radio Base Stations
	d_{RBS}	m	Dimensions of an RBS cell
	D_{net}	s	Cumulative network delay
Data Centre	N_{DC}		Number of Data Centres
	N^i_S, where $i = 1,2,\ldots,N_{DC}$		Number of servers in Data Centre
	N^j_{CPU}, where $j = 1,2,\ldots,N^i_S$		Number of CPUs per server
	s^j_{CPU}, where $j = 1,2,\ldots,N^i_S$	MIPS	CPU's speed
	$memory^j$, where $j = 1,2,\ldots,N^i_S$	MB	Amount of memory per server
	$storage^j$, where $j = 1,2,\ldots,N^i_S$	GB	Amount of storage per server
	$network^i_{bw}$, where $i = 1,2,\ldots,N_{DC}$	Mb/s	Network bandwidth
	t_{init}, t_{idle}, t_{term}	s	Times of VM transitions
Service Placement	$placement = \{every, n\text{-}closests\}$		Service placement policy
OBJECTIVES			
Quality of Service	RT^i, where $i = 1,2,\ldots,N_{ser}$	s	Application response time
	TP^i, where $i = 1,2,\ldots,N_{ser}$	req/s	Application throughput
Costs	$Cost$	$	Total costs of infrastructure

homogeneous mobile network composed of N_{RBS} equidistantly distributed RBSs.

In the absence of a specific mobile generational standard, an MD is handed over between RBSs at the geographic point where they cross the cell boundary distinguishing two independent RBSs defined by the width of the rectangular cells d_{RBS}.

Data Centre

The DC model captures the influence that its capacity has on performance and costs of computation, as described in Section 4.

To capture the influence on performance, quantity and quality of each DC resource is described. DC i consists of N^i_S servers, that can differ in specification. Server j contains N^j_{CPU} CPUs capable of executing s^j_{CPU} operations in every second. Values of $memory^j$ and $storage^j$ specify the total amount of available memory and storage, respectively. The network bandwidth is specified with $network^i_{bw}$. The DC model includes also a provisioning model, that describes how available resources are shared among several applications, e.g., time-sharing or space-sharing.

A DC hosts services in VMs. A service can be distributed over multiple VMs. Incoming workload is load-balanced by either a method of round-robin, random selection, or placed in the VM with the lowest load. However, a user requests are always forwarded to the VM that served his first request. A service can specify a minimum and maximum number of VMs it requires. The DC scales the application within these bounds based on the load-balancing outcome.

To emulate the life-cycle of a VM we have defined six VM states that are described in Table 3. The transitions between the states are as presented in Figure 4.

At the beginning all VMs are in INACTIVE state. A VM is initiated when the first request arrives to a DC. It takes t_{init} seconds before VM is ready to start

Table 3: States of Virtual Machine.

Name	Description
INACTIVE	VM is turned off.
INITIATING	VM is booting up.
PROCESSING	VM is serving requests.
IDLE	VM is waiting for requests.
MIGRATING	VM is transmitting data.
TERMINATING	VM is shutting down.

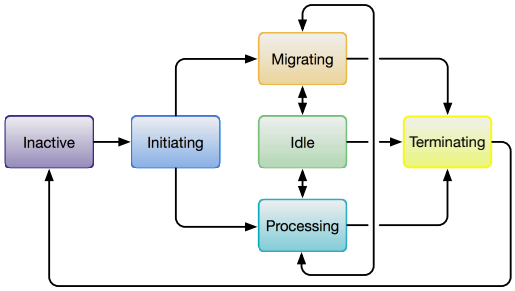


Figure 4: Transitions between Virtual Machine states.

processing requests or receiving migrated requests and user state from other DC. We assume that a VM is not able to process requests and handle migrations at the same time, so it changes state between PROCESSING and MIGRATION over the time. In our implementation, migrations have higher priority than processing, so if there are any migrations to handle processing is paused. When there are no requests to process and no migrations to handle a VM goes into IDLE state. A VM is terminated if IDLE state lasts for longer than t_{idle} seconds, and the VM termination takes t_{term} seconds. In the ideal case, a VM spends most of its time in the PROCESSING state and other, non-revenue generating, states are minimised. MIGRATION state is directly associated with the overhead of dispersing computational resources and the mobility of end users.

Service Placement

Service placement policies define in what DC(s) a service should be hosted, what number of replicas should be running, and when a service should be migrated from one DC to another. These decisions depend on the mobility of users, the size of users' state that has to be migrated following the users' movement, and QoS requirements. For example, a service can be hosted in n proximal DCs closest to the majority of its users (*n-closests*), or in the case of latency sensitive services in every Proximal DC that is needed to provide acceptable QoS (*every*).

6.3 Objectives Model

The third group of parameters in Table 2 describes QoS and costs of the telco cloud.

Quality of Service

Combining the resource requirements model, which describes the amount of resources an application needs, with a DC model, allows to simulate how collocation of different services in a DC influences their response times RT^i and throughputs TP^i.

Costs

In our opinion the cost models available in the literature and described in Section 5.3 are very "country dependent", because of the inclusion of variable parameters such as salaries, costs of energy or costs of property. They are also not taking into account parameters important from the perspective of the telco cloud, such as the size of DC. Therefore, we model the costs of the telco cloud using a basic heuristic based on observation that dispersion of infrastructure causes additional costs, e.g.: increase of administrator travel time between locations, and higher unit costs of computation in proximal DCs because of smaller scale and high initial costs.

$$\text{Cost} \propto \frac{N_{\text{DC}}}{\sum^{\text{DC}} N_S} \tag{1}$$

As shown in Equation 1, the total cost of a telco cloud is directly proportional to the number of DCs and inversely proportional to the total number of servers in all DCs. It means that distributing the same number of servers among many DCs is more expensive than placing them in one DC.

6.4 Limitations

The proposed meta-model has several limitations. The application model assumes that all requests generated by one application are homogeneous and each of them consumes the same amount of resources. The mobile access network model does not take into account the physical layer, channel provisioning, and cell load balancing. Additionally, the radio access network functions as a mechanism to associate MDs with DCs propagation and system processing delays are thus not modelled.

7 SIMULATION SHOWCASE

To demonstrate the scope of the telco cloud meta-model and the simulator that we have implemented we introduce an elementary showcase scenario below. The scenario is designed to reveal the basic relationship between workload – MD mobility, setup – Proximal DC catchment, and objectives – the aggregate utilisation of a telco cloud.

We have implemented a coarse grained simulator using SimJava (Howell and McNab, 1998) as the underlying event-driven simulation framework. All modules are implemented from scratch but are based on the meta-model presented in Section 6. The simulator fully implements the proposed request generation and network models, but has implemented more abstract mobility, resource requirements, DC, and service placement models.

7.1 Experiments

For the sake of simplicity we decided to present a simplified scenario. Only one service in considered and the size of simulation is reduced significantly, comparing to the desired scale. The VM scalability and placement models are just a proof-of-concept. The goal is to obtain clear conclusions about the relationship between MD mobility and DC catchment, and avoid the interference of other elements. The scenario is described in detail below.

The telecommunication infrastructure is composed of 16 RBSs, in a 4x4 layout, as presented in Figure 5. The cells are tangent but not overlapping and are dimensioned as a typical LTE micro-cell at 750 m, as detailed in (Shahab et al., 2013). The number of DCs is varied in the experiments and thus so also the DC catchment (ratio between DCs and RBSs) varies between (1 : 1) and (1 : 16), see Figure 5. In abstract terms, the (1 : 1) catchment case represents a setup with one Proximal DCs per RBS. In contrast, the (1 : 16) catchment case approaches a more traditional Remote DC serving all users in the domain.

To reveal the effects of DC catchment, all DCs are of the same capacity. The number of VMs in each DC is scaled proportionally to the number of users they serve. The DC in the (1 : 16) catchment scenario has 16 VMs, while the DC in the (1 : 1) scenario has just one VM. The workload is balanced among available VMs, new sessions are forwarded to the least loaded VM. To reveal the full extent of the effect of user mobility, user states and requests are strictly migrated to the geographically nearest DC.

We use a request generation model with a session arrival rate of λ_{ses} described by a Log-Normal

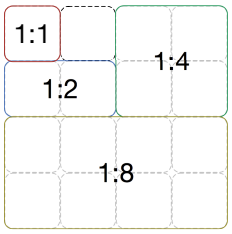


Figure 5: DC catchment scenarios. Solid lines represent the DC catchment (1 : 16 case covers whole simulation area with one DC).

distribution with the parameters $\mu = 3$ and $\sigma = 1.1$. The number of requests per session N_{req} is taken from an Inverse Gaussian distribution with the parameters $\lambda = 5$ and $\mu = 3$. Inter-request time is D_{req} seconds and is modelled with with an Exponential distribution with $\lambda = 0.1$. The simulation domain is populated by 480 MDs, all subscribing to the same service. Due to the size and simplicity of the network topology in the proposed scenario, we are deploying a Markov based mobility model. The mobility mode is based on a car and is as specified in Section 6.1, with parameters from (Bettstetter, 2001). To allow the mobility and workload models to jointly reach a steady state, the simulation is run for 8 simulated hours. This results in an average processing load of 30%, this level should give enough margin to for example migrations to complete successfully.

In order to investigate the influence of Proximal DC catchment on the aggregate performance of the telco cloud we observe the life cycle of the VMs that run within the DCs by recording the amount of time each VM spends in each state.

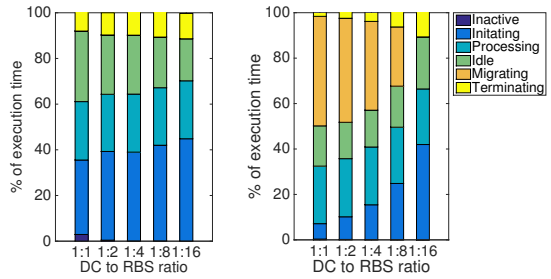
The user state is proportional to the aggregate size of that user's sessions with the application it subscribes to and is defined by a 5^{th} order AR-process with linearly decaying parameters. In our simulation, initialisation of a VM takes $t_{\text{init}} = 81s$, similarly as for m1.small VM type in Amazon EC2 (Ostermann et al., 2010). A VM is terminated if it remains in the IDLE state longer than t_{idle} which is equal to the mean inter-session time. It takes $t_{\text{term}} = 21s$ to terminate a VM.

We run two sets of experiments. In the first set, end users are static. The second set of tests introduces mobility. In both sets we investigate the variations in the distribution of time that VMs spend in each state.

7.2 Results

Figure 6 shows the breakdown of the mean time spent in each VM state in the system per DC catchment. With a (1 : 1) DC catchment the utilisation suffers from the proportion of time spent in IDLE state due to the relatively low request arrival rate generated by

one sixteenth of all users. The inefficiency is caused by the time the system spends in the IDLE, INITIATING, and TERMINATING states. The composition of time spent in these states changes with DC catchment, and is a reflection of the number of VMs in a DC and load-balancing effort. Reducing the time spent on starting and terminating VMs would free up more resources and perhaps also make the system more reactive to sudden workload changes. The intelligent management of VM scalability and placement is clearly something that needs to be optimised.



(a) Static users (b) Mobile users

Figure 6: DC catchment vs. time spent in each VM state.

Figure 6(b) reveals the overhead of user mobility and the migration effort it incurs. Depending on the DC catchment, different migration dynamics come into play. As migrations are more frequent in the $(1:1)$ case than in the $(1:8)$ case, user states do not have the time to grow as much between migrations in the former case. The migration effort is therefore not a factor eight lower in the $(1:8)$ case versus the $(1:1)$ case, but rather, they spend 26% and 47% of their time in the MIGRATING state, respectively. The system dynamics revealed by Figure 6(b), where at worst, 47% of the execution time is spent migrating users, points to the need to find scaling mechanisms for the telco cloud that take into account mobility and inactivity, so that resources can be freed dynamically for other revenue generating applications. A policy of strictly migrating user states and requests to the geographically closest DC, irregardless of DC catchment, in order to obtain minimal propagation and communication latency, is suboptimal.

8 FUTURE WORK AND CONCLUSIONS

In this paper we present a way to combine existing models of user mobility, mobile and core networks, and DCs into a meta-model that is capable of capturing dynamics of the telco cloud. We also implement a prototype simulator based on simplified meta-model.

The meta-model can be used by telecommunication operators as well as equipment developers to model an existing infrastructure and to plan future changes. Researchers can test new algorithms for resource management, e.g., migration of services between geographically distributed DCs. Also developers of mobile applications can benefit from using the simulator to observe how their applications behave in telco cloud environment.

Future work will be focused on enhancing the functionality of the simulator to incorporate other parameters from the presented meta-model. Then, using the simulator we would like to explore the following telco cloud challenges: minimising the trade-offs between costs and performance of telco cloud depending on the placement and capacity of DCs, and optimal placement and preemptive/predictive migration of services between DCs.

ACKNOWLEDGEMENTS

This work is funded by the Swedish Research Council (VR) project Cloud Control and the European Union's Seventh Framework Programme under grant agreement 610711 (CACTOS).

Maria Kihl and William Tärneberg are members of the Lund Center for Control of Complex Engineering Systems (LCCC) funded by the Swedish Research Council. Also, they are members of the Excellence Center Linköping - Lund in Information Technology (ELLIIT).

REFERENCES

Abhari, A. and Soraya, M. (2010). Workload generation for youtube. *Multimedia Tools and Applications*, 46(1):91–118.

Ahmed, A. and Sabyasachi, A. S. (2014). Cloud computing simulators: A detailed survey and future direction. In *2014 IEEE International Advance Computing Conference (IACC)*, pages 866–872. IEEE.

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

Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer networks*, 54(15):2787–2805.

Barker, S. K. and Shenoy, P. (2010). Empirical evaluation of latency-sensitive application performance in the cloud. In *Proceedings of the first annual ACM SIGMM conference on Multimedia systems*, pages 35–46. ACM.

Baroncelli, F., Martini, B., and Castoldi, P. (2010). Network virtualization for cloud computing. *Annals of telecommunications – Annales des télécommunications*, 65(11-12):713–721.

Bettstetter, C. (2001). Smooth is better than sharp: A random mobility model for simulation of wireless networks. In *Proceedings of the 4th ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, MSWIM '01, pages 19–27, New York, NY, USA. ACM.

Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, pages 13–16, New York, NY, USA. ACM.

Bosch, P., Duminuco, A., Pianese, F., and Wood, T. L. (2011). Telco clouds and virtual telco: Consolidation, convergence, and beyond. In *2011 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 982–988. IEEE.

Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A. F., and Buyya, R. (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, 41(1):23–50.

Dinh, H. T., Lee, C., Niyato, D., and Wang, P. (2013). A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing*, 13(18):1587–1611.

Drolia, U., Martins, R., Tan, J., Chheda, A., Sanghavi, M., Gandhi, R., and Narasimhan, P. (2013). The case for mobile edge-clouds. In *Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC)*, pages 209–215.

Fehske, A., Richter, F., and Fettweis, G. (2009). Energy efficiency improvements through micro sites in cellular mobile radio networks. In *GLOBECOM Workshops, 2009 IEEE*, pages 1–5.

Greenberg, A., Hamilton, J., Maltz, D. A., and Patel, P. (2008). The cost of a cloud: Research problems in data center networks. *SIGCOMM Comput. Commun. Rev.*, 39(1):68–73.

Howell, F. and McNab, R. (1998). Simjava: A discrete event simulation library for java. *Simulation Series*, 30:51–56.

Kliazovich, D., Bouvry, P., and Khan, S. (2012). Greencloud: a packet-level simulator of energy-aware cloud computing data centers. *The Journal of Supercomputing*, 62(3):1263–1283.

Kovachev, D. (2012). Framework for computation offloading in mobile cloud computing. *International Journal of Interactive Multimedia and Artificial Intelligence*, 1(7):6–15.

Lee, J. J. and Gupta, M. (2007). A new traffic model for current user web browsing behavior. *Intel Corp.*

Li, F., Dustdar, S., Bardram, J., Serrano, M., Hauswirth, M., Andrikopoulos, V., and Leymann, F. (2013). Eupaas-elastic ubiquitous platform as a service for large-scale ubiquitous applications. In *Proceedings of the 3rd International Conference on Cloud Computing and Service Science, CLOSER 2013*, pages 309–314.

Meisner, D., Wu, J., and Wenisch, T. F. (2012). Bighouse: A simulation infrastructure for data center systems. In *2012 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 35–45. IEEE.

Ostermann, S., Iosup, A., Yigitbasi, N., Prodan, R., Fahringer, T., and Epema, D. (2010). A performance analysis of ec2 cloud computing services for scientific computing. pages 115–131. Springer.

Papagiannaki, K., Moon, S., Fraleigh, C., Thiran, P., and Diot, C. (2003). Measurement and analysis of single-hop delay on an IP backbone network. *IEEE Journal on Selected Areas in Communications*, 21(6):908–921.

Patel, C. D. and Shah, A. J. (2005). Cost model for planning, development and operation of a data center.

Patel, M., Naughton, B., Chan, C., Sprecher, N., Abeta, S., and Neal, A. (2014). Mobile-edge computing – introductory technical white paper. Technical report.

Reyes-Lecuona, A., González-Parada, E., Casilari, E., Casasola, J., and Diaz-Estrella, A. (1999). A page-oriented www traffic model for wireless system simulations. In *Proceedings ITC*, volume 16, pages 1271–1280.

Riley, G. F. and Henderson, T. R. (2010). The ns-3 network simulator. In *Modeling and Tools for Network Simulation*, pages 15–34. Springer.

Sakellari, G. and Loukas, G. (2013). A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing. *Simulation Modelling Practice and Theory*, 39(0):92 – 103.

Shahab, S. N., Kiong, T. S., and Abdulkafi, A. A. (2013). A framework for energy efficiency evaluation of lte network in urban, suburban and rural areas. *Australian Journal of Basic and Applied Sciences*, 7(7):404–413.

Varga, A. et al. (2001). The OMNeT++ discrete event simulation system. In *Proceedings of the European simulation multiconference (ESM'2001)*, volume 9, page 65.

Wang, A., Iyer, M., Dutta, R., Rouskas, G. N., and Baldine, I. (2013). Network virtualization: Technologies, perspectives, and frontiers. *Journal of Lightwave Technology*, 31(4):523–537.

Weingartner, E., vom Lehn, H., and Wehrle, K. (2009). A performance comparison of recent network simulators. In *2009. ICC '09. IEEE International Conference on Communications*, pages 1–5.

Wickremasinghe, B., Calheiros, R., and Buyya, R. (2010). Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. In *2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, pages 446–452.