2011 IEEE International Conference on Cluster Computing

Large-Scale Simulator for Global Data Infrastructure Optimization

Sergio Herrero-Lopez, John R. Williams and Abel Sanchez
Intelligent Engineering Systems Laboratory
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139
Email: $\{sherrero, jrw, doval\}$ @$mit.edu$

***Abstract*—IT infrastructures in global corporations are appropriately compared with nervous systems, in which body parts (interconnected datacenters) exchange signals (request-responses) in order to coordinate actions (data visualization and manipulation). A priori inoffensive perturbations in the operation of the system or the elements composing the infrastructure can lead to catastrophic consequences. Downtime disables the capability of clients reaching the latest versions of the data and/or propagating their individual contributions to other clients, potentially costing millions of dollars to the organization affected. The imperative need of guaranteeing the proper functioning of the system not only forces to pay particular attention to network outages, hot-objects or application defects, but also slows down the deployment of new capabilities, features and equipment upgrades. Under these circumstances, decision cycles for these modifications can be extremely conservative, and be prolonged for years, involving multiple authorities across departments of the organization. Frequently, the solutions adopted are years behind state-of-the-art technologies or phased out compared to leading research on the IT infrastructure field. In this paper, the utilization of a large-scale data infrastructure simulator is proposed, in order to evaluate the impact of "what if" scenarios on the performance, availability and reliability of the system. The goal is to provide datacenter operators a tool that allows understanding and predicting the consequences of the deployment of new network topologies, hardware configurations or software applications in a global data infrastructure, without affecting the service. The simulator was constructed using a multi-layered approach, providing a granularity down to the individual server component and client action, and was validated against a downscaled version of the data infrastructure of a Fortune 500 company.**

***Keywords*-computer simulation; system modelling; IT infrastructure;**

I. Introduction

A. Motivation

The relevance of information technology infrastructures in corporations has grown in consonance with the unstoppable phenomenon of globalization. Multinational corporations have expanded their presence across multiple continents to offer their services or products directly to every region while capturing local talent and resources. Nevertheless, these distributed corporations still operate as integral units thanks to the interconnectivity provided by global data infrastructures, which have been appropriately compared to nervous systems [1][2]. In this scenario, globally distributed organizations never sleep, having personnel visualizing, generating, manipulating and sharing information assets throughout all time zones. This modus operandi was never exclusive of telecom or web companies, in which the infrastructure itself represents the core business; but global collaboration also became key for a wide variety of other organizations in which IT has become more than an indispensable tool. Today, banking, pharmaceutical or automobile industries cannot properly function without a platform and a set of tools that enable them creating, visualizing and manipulating information across remote locations. For these reasons, performance, reliability and availability of latest data on these infrastructures are mayor concerns for these organizations, and getting them right for a low cost has become a key differentiating factor against the competition.

Data sharing and collaboration capabilities have given global organizations the flexibility, agility, robustness and efficiency to operate without pause. However, these advantages have also lead to an unprecedented dependency on IT infrastructures. Almost without exception, downtime is considered unaffordable and oftentimes the performance of the system and the availability of fresh information are sacrificed, so as to keep the system operative. Unfortunately, a fully operative infrastructure cannot be left "as is" and there are two motivations that drive change: 1) the integration of new features, state-of-the-art technologies or improved practices is necessary in order to maintain a competitive edge. 2) The need to cut costs by reducing the complexity of the architecture and maintenance of the system. Under these circumstances, decisions susceptible of affecting IT operations need to go through exhaustive reviewing processes across individuals, groups and divisions of the corporation so as to minimize the risk of stopping the natural flow of information. Hence, it is not surprising that non-critical features, cutting-edge technologies and latest software updates or protocols, are not taken into production immediately for the sake of preserving the *status quo*, unless is strictly necessary.

In this paper, the construction of a global data infrastruc-

978-0-7695-4516-5/11 $26.00 © 2011 IEEE
DOI 10.1109/CLUSTER.2011.15

ture simulator is presented, in order to evaluate the impact of "what if" scenarios on the performance, availability and reliability of large-scale systems. The simulator takes as input the workload of each application, the resources allocated by individual user requests, the network topology of the organization, the hardware configuration deployed in each datacenter and details on background processes. Using this information, the queueing network models that integrate the simulator produce estimates of the response time for each user request, along with measurements of the resource allocation and network utilization, so as to facilitate optimization goals for datacenter operators.

The information generated by the simulation platform can be used towards diverse optimization purposes: 1) *Performance estimation*, which enables the response time to be evaluated for a given workload, network topology, hardware configuration and software application. 2) *Capacity planning*, which enables the datacenter to determine the resources required to meet Service Level Agreements (SLA). 3) *Hardware/Software configuration*, which enables calibrating both hardware and software parameters to achieve optimum performance and utilization goals. 4) *Network administration*, which allows the topology of the global network to be designed to cope with the expected traffic. 5) *Bottleneck detection*, which enables potential infrastructure bottlenecks to be identified and prevented. 6) *Background job optimization*, which facilitates the scheduling and effectiveness of jobs such as replication or indexing. 7) *Internet Attack protection*, which allows evaluating mechanisms to thwart denial-of-service attacks.

B. Related Work

Latest work on understanding and optimizing IT infrastructures can be divided into three groups: *Analytic Modeling*, *Profiling* and *Simulation*.

Initially, *analytic models* for different datacenter tiers, such as application servers, web servers o database servers were constructed separately. Abdelzaher et al. utilize classical feedback control theory to model web server systems and provide performance guarantees [3]. An application server supporting an e-commerce web portal is modeled by Villela et al. as a $M/G/1/PS$ queuing system [4]. Ahmad et al. use statistical modeling to understand the interaction between concurrent queries on a database server [5]. Recent initiatives construct models that combine different tiers and represent entire datacenters. Urgaonkar et al. present a multi-tier model based on networks of queues, where each queue represents a tier. They provide a general model capable of representing the behavior of different tiers with different responsibilities and performance metrics [6]. Additional factors that appear in multi-tier scenarios such as, sessions, caching between tiers and concurrency limits, are also included in their model. Nevertheless, their model does not provide granularity down to the server or hardware component level. Urgaonkar et al. take this work a step further and describe predictive and reactive methods that determine the capacity to add to the datacenter using the same queuing model [7].

Profiling in a datacenter can be carried out at different levels. Yu et al. constructed a network application profiler, called *SNAP*, which not only identifies performance problems but also identifies their causes [8]. The profiler system collects TCP statistics and socket logs, and correlates these across shared resources. Ren et al. describe the web-scale profiling mechanism on Google's infrastructure [9], known as Google-Wide Profiling (GWP). This work not only measures network performance, but also takes samples across machines distributed in datacenters and gathers application-specific data. It pays special attention to the challenges arisen by profiling at large scale. It is also necessary to mention the work described by Gmach et al. on profiling power consumption in datacenters [10]. Improving energy management in IT infrastructures is critical not only for cost reduction but also for the environment.

Finally, *simulation* platforms are also constructed to reproduce the behavior of different elements within an IT infrastructure. Ellithorpe et al. constructed an FPGA-based simulator that enabled experimenting with large-scale datacenter network architectures [11]. A general and flexible multi-tier datacenter simulation platform, called *MDCSim*, is capable of predicting performance for different tiers, with different hardware configurations and connected through different networks [12]. Finally, in the context of cloud computing, tools that simulate the effect of adding virtual resources to datacenters were presented by the *CloudSim Toolkit* [13].

C. Research Contributions

In this paper a global data infrastructure is modeled and simulated. The simulator is designed to handle globally distributed datacenters in which clients in different time zones visualize, manipulate and transfer data concurrently using a variety of software applications. In this subsection the differences between previous research and this work are emphasized.

- *Global Infrastructure:* As opposed to other analytic models and simulations preceding this work which only considered a single datacenter [6] [12], this paper presents a simulator platform for an entire global infrastructure composed by multiple multi-tier datacenters connected through networks across continents.
- *Application Diversity:* As opposed to other research which only considered one or few applications, this work proposes a model to represent any software application and provides the capability of intertwining multiple workloads. Each application is modeled as a series of client operations, which in turn are decomposed into trees of messages. These messages

flow concurrently through the infrastructure allocating hardware resources.

- *Background Jobs:* As opposed to other initiatives that focus exclusively on user centered applications [6] [12], this work enables simulating background processes, such as replication or indexing, running simultaneously with user generated workloads.
- *Simulator Validation:* The accuracy of this simulator was validated against the global IT infrastructure of a Fortune 500 company running three applications: Computer-Aided Design (CAD), Visualization (VIS) and Product Data Management (PDM). This infrastructure was composed by multiple datacenters, with thousands of clients visualizing and manipulating files, while replication and indexing jobs were carried out simultaneously in the background. To the best of our knowledge, this research pioneers the simulation of interconnected datacenters validated with data collected from the infrastructure of a Fortune 500 company.

Modeling and simulation of IT infrastructures has two main advantages compared to the infrastructure profiling approach:

1) *Simplicy & Cost:* Profiling datacenter behavior requires running processes that measure vast amounts of highly detailed information. As the infrastructure size increases the profiling overhead can degrade system performance and leaving collected data unexploited, unless additional resources are allocated, which increases the overall profiling cost. The simulation platform proposes a simpler yet powerful non-intrusive tool capable not only of reproducing system behavior on a high level, but also predicting the impact of "what if" scenarios for a lower cost.
2) *Consensus Seeker:* The high complexity of a global infrastructure makes it impossible for a single datacenter operator to understand all the dynamics of the workload, hardware, software or network; in fact, typically individuals do not have detailed knowledge about the system beyond their area of responsibility. The simulation platform serves as a unique perspective that enables decision makers being on the same page and reaching consensus on system alterations.

This publication is organized as follows: Section II presents the principles on which our global data infrastructure simulator is built upon, along with the details on the queueing network models utilized to represent hardware and the message tree representation for software. Section III includes the set of experiments utilized to validate the simulator against a downscaled version of the IT infrastructure of a Fortune 500 company using profiling data collected in their laboratory. Section IV includes a case study with predictions generated by the simulation of the infrastructure of this global company, but introducing alterations aiming to reduce costs, while keeping the same quality of service. Finally, section V gathers the conclusions derived from this work.

II. Global Data Infrastructure Simulator

A. Multi-Agent System (MAS)

The simulator platform is constructed following a multi-layered approach formed by *Components* and *Operations*.

Components are stateful autonomous agents that represent parts of the infrastructure at various granularities. These components interact with each other through message exchange: they accept input messages of pre-established types that alter their internal states, and based on this state they produce output messages addressed to other components. High-level components can stand for datacenters or network links. These include medium-level components such as application server tiers or database server tiers. Servers, in turn, are modeled using low-level components such as memory, processors, disk arrays or network cards. The modularity of this design grants the flexibility to add/remove or reutilize components. Specific behaviors associated to each type of component can be adjusted to the real hardware by tuning its configuration parameters. These behaviors are explained in subsection II-B. A basic example of a global IT infrastructure with different component granularities is illustrated in figure 1. This infrastructure is composed by a *master* datacenter component, D_{NA}, and multiple *slave* datacenter components (D_{EU}, D_{AS}, D_{SA}, D_{AFR}, D_{AUS}) connected through network switches (sw). Clients, c, are served by their closest datacenter. The master datacenter hosts four tier components - application server tier (T_{app}), database server tier (T_{db}), file server tier (T_{fs}) and index server tier (T_{idx}) -, while slave datacenter components only host T_{fs}. T_{fs} and T_{db} tiers are backed by their respective Storage Area Networks (san). Individual server components are composed by low-level components that represent hardware resources: memory (mem), processor (cpu), network card (nic) and disk arrays ($raid$).

Operations define the interactions between clients distributed across continents and the software application running on the infrastructure. Examples of typical operations are LOGIN, SEARCH or OPEN. Each application consists of a set of client-initiated operations. Background jobs are also represented as operations, but are periodically scheduled by the datacenter instead. Operations are decomposed into sequences of messages that flow between datacenters, server tiers and hardware components affecting their internal status. Each individual message will contain information about the source and target component, along with the resources needed to process it. Based on the current state of the target component, its specifications and resources required by the message, the processing time will be calculated. The cumulative processing time of the sequence of messages pertaining to the same operation yields the total operation

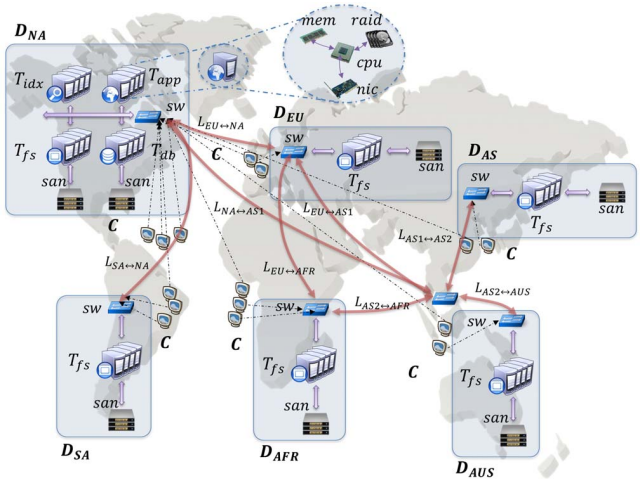

Figure 1. Example of a Global IT Infrastructure represented by simulator components

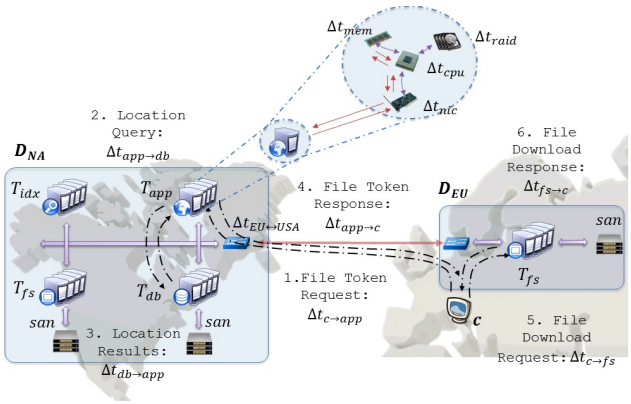

Figure 2. Operation decomposition for the OPEN operation

response time. Figure 2 illustrates the decomposition of a classic file opening (OPEN) operation into a sequence messages. First, the client c in EU makes a request to T_{app} in the master datacenter, D_{NA} for the token needed to download the latest version of a file from the T_{fs} in D_{EU}. The T_{app} checks for metadata about this file in the T_{db} to make sure the T_{fs} in D_{EU} has indeed the latest version, if not, a synchronization request between the T_{fs} in D_{NA} and the T_{fs} in D_{EU} would be triggered through the network link $L_{EU\to NA}$. Upon token reception by c, the token is used to download the file directly from T_{fs} in D_{EU}. The total response time of the OPEN operation is calculated by adding the time measured at each step, as shown in equation 1.

$$t_{OPEN} = \Delta t_{c\to app} + \Delta t_{app\to db} + \Delta t_{db\to app} + \Delta t_{app\to c} + \Delta t_{c\to fs} + \Delta t_{fs\to c} \quad (1)$$

Each step is further decomposed into the messages exchanged by low-level components. For example, the token request from c to T_{app}:

$$\Delta t_{c\to app} = \Delta t_{EU\to NA} + \Delta t_{nic} + \Delta t_{cpu} + \Delta t_{raid} \quad (2)$$

The processing time of an individual message going through a low-level component depends on its internal state, which is affected by messages originated by other clients being processed simultaneously at the component.

The collection of components exchanging messages constitutes a Multi-Agent System (MAS) that reproduces the behavior of a global data infrastructure. Recently, MAS have been successfully utilized to simulate self-organizing computer networks [14], Service Oriented Architectures (SOA) [15] and High Performance Computing (HPC) systems [16]. To the best of our knowledge, this is the first time a MAS is used to simulate an IT infrastructure composed by several globally distributed multi-tier datacenters.

B. Hardware modeling

In this subsection the behavior of the *components* that the MAS is built upon is described. Consider an infrastructure with K datacenters denoted by D_k with $k = 1 \dots K$. Datacenters distributed across continents are interconnected through network links $L_{D_i\to D_j}$ where $i \neq j$. Each facility D_k runs $M^{(k)}$ tiers denoted by $T_m^{(k)}$ with $m = 1 \dots M^{(k)}$. Each tier $T_m^{(k)}$ is formed by $N^{(k,m)}$ servers of the same type denoted by $S_n^{(k,m)}$ with $n = 1 \dots N^{(k,m)}$. Similarly, servers within a datacenter are interconnected through local network links, $L_{S_n^{(k,m)}\to S_q^{(k,p)}}$ where $n \neq q$. The collection of hardware components in each server $S_n^{(k,m)}$ are modeled by a set of $P^{(k,m,n)}$ interconnected queues denoted by $Q_p^{(k,m,n)}$ with $p = 1 \dots P^{(k,m,n)}$. One component can be represented by a single queue or a queueing network, and its internal structure and service rates are established by the technical specifications of the hardware being modeled. Each D_k gives service to a group of $Z^{(k)}$ clients, C_z with $z = 1 \dots Z^{(k)}$, and just like servers, are also modeled as networks of queues. Queueing network models have been extensively applied as powerful tools for performance modeling, evaluation and prediction of computer systems [17][18]. Next, the models standing for low-level hardware components are introduced using Kendall's three-factor $A/B/C$ notation system. Queues can be of types First Come First Served (FCFS) or Processor Sharing (PS).

- *CPU*: CPUs are modeled as $M/M/s$ FCFS queues. s is the number of cores in the CPU. Servers powered by multiple sockets are represented by k-$M/M/s$ FCFS queues [19]. The CPU component is fully specified by providing the number of sockets, number of cores per socket and the frequency of each core (GHz).
- *Memory*: For a message in the CPU queue to be processed, it is necessary to have enough memory available. Otherwise, the message waits in the queue until enough memory is released.

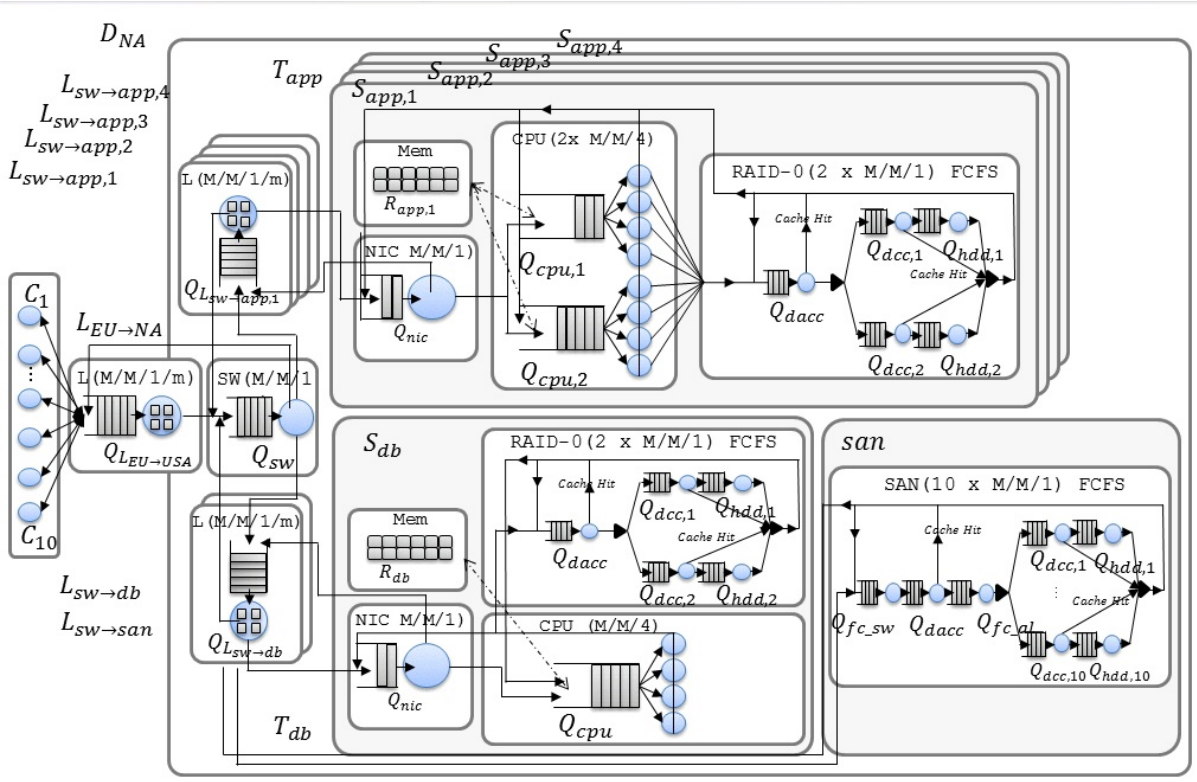

Figure 3. Modeling a multi-tier datacenter using networks of queues and links

- *NIC*: Network Interface Cards (NIC) enable communication with other servers in the tier and are modeled as $M/M/1$ FCFS queues. The NIC component is fully specified by its speed (Mbps) [20].
- *Switch*: Network switches bring together all the tiers of the datacenter and just like the NICs are represented by $M/M/1$ FCFS queues. The network switch is fully specified by its speed (Mbps) and is typically orders of magnitude higher than NICs speed [20].
- *Links*: Network links between datacenters and links between machines within a datacenter are modeled by $M/M/1/r$ PS queues, where r is the maximum number of connections [20]. The model is fully specified by the bandwidth (Mbps), the maximum number of connections and the latency (ms).
- *RAID*: Read/Writes to disk arrays composed by d disks are modeled as d-$M/M/1$ FCFS queues on a fork-join configuration preceded by an $M/M/1$ FCFS queue representing the RAID controller [21] [22] [23]. The model is fully specified by the number of disks d, the disk data transfer rate (MB/s), the disk controller speed (Gbps), the disk controller cache (MB), the disk array controller speed (Gbps) and the disk array controller cache (GB).
- *SAN*: Read/Writes to a Storage Area Network composed by d disks are also modeled as d-$M/M/1$ FCFS queues on a fork-join configuration preceded by three $M/M/1$ FCFS queues representing a fiber channel switch, a disk array controller and cache, and a fiber channel arbitrated loop [24]. The SAN is connected to other servers in the datacenter through a network link. The model is fully specified by the number of disks d, the disk data transfer rate (MB/s), the disk controller speed (Gbps), the disk controller cache (MB), the fiber channel switch speed (Gbps), the disk array controller speed (Gbps), the disk array controller cache size (GB), the fiber channel arbitrated loop speed (Gbps).

Figure 3 illustrates an example of a datacenter in the USA with two tiers, application tier T_{app} and database tier T_{db}, connected through a switch Q_{sw}. T_{app} is composed by four servers $S_{app,i}$ with $i = 1 \ldots 4$. Each $S_{app,i}$ contains a network card Q_{nic}, a dual socket quad-core CPU $(Q_{cpu,1}, Q_{cpu,2})$ and a RAID $Q_{raid_c}, (Q_{dcc,i}, Q_{hdd,i})$ with $i = 1 \ldots 2$. T_{db} is composed by a single server S_{db} and a san. S_{db} has a network card Q_{nic}, a quad-core CPU Q_{cpu} and an identical RAID. san is formed by a fiber channel switch Q_{fc_sw}, a disk array controller and cache Q_{dacc}, a fiber channel arbitrated loop Q_{fc_al} and an array of disks $(Q_{dcc,i}, Q_{hdd,i})$ with $i = 1 \ldots 10$. Clients c_z with $z = 1 \ldots 10$ are connected to the datacenter through link

$L_{EU \to NA}$, and the servers and the SAN are connected to the switch through local network links $L_{sw \to S_{app,i}}$, $L_{sw \to S_{db}}$ and $L_{sw \to san}$.

In summary, the multiple layers of hardware supporting the infrastructure are modeled as queueing networks interconnected by links, which altogether yield the behavior of the MAS. The total duration of a request is obtained by accumulating the processing/waiting times (queues) and transfer times (links).

C. Application modeling

In this subsection the model utilized to represent a distributed software application is explained. As mentioned in subsection II-A, each application is divided into a collection of *operations* that clients launch following a daily workload. The application model has two elements:

- *Application Workload*: The application workload registers the number of clients that launch an operation by location and time of the day, along with the distribution of the operation types and its fluctuation throughout the day.
- *Message Tree*: Every operation is modeled as a tree of messages that is generated when a client request is served by the infrastructure. Each message in the tree represents an interaction between components in the system and encapsulates information about the associated resource allocation and processing cost. This information is reflected by four hardware-agnostic parameters: Data transfer in KB (R_t), memory footprint in KB (R_m), processing cost in thousands of cycles (R_p) and disk read/write in KB (R_d). These parameters dictate the work to be carried out in queueing networks and/or links, and hence, determine the final duration of the operation. The message tree dictates the types of tiers involved, however, the exact datacenter, server and hardware instances are decided runtime by the simulator, based on the input workload and load-balancing strategies.

Figure 4 illustrates the message tree for an OPEN operation. Segment (1) represents the client query to T_{db} via T_{app} to obtain a token to download the latest version of the desired file. Using this token, Segment (2) represents the download of the desired file from T_{fs}. The fist message of the tree reflects an interaction between one client c and one server in T_{app}. As indicated by the (R_t, R_m, R_d, R_p) parameter set, this message involves transfering 30 KB across a network link, allocating 5 MB of main memory, consuming 500 thousand cycles on a single CPU core and reading 3 MB from the SAN.

III. Model Validation

The main inconvenient that prevents medium size organizations from implementing global infrastructure-wide profiling mechanisms is the high cost and the incapacity to predict

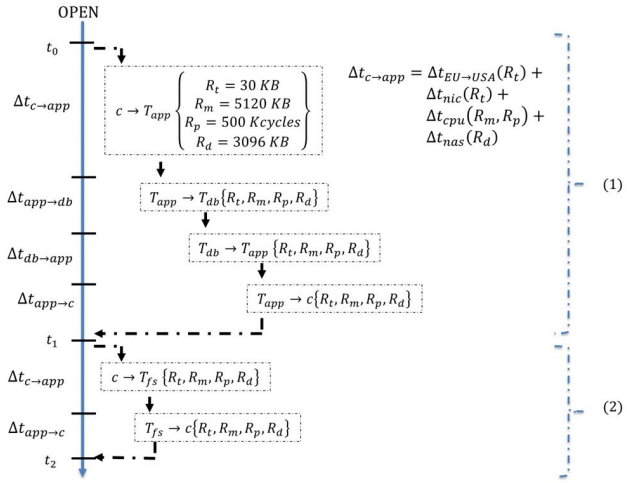

Figure 4. OPEN Operation Definition

the behavior of the system when parameters are modified. Nevertheless, companies still maintain dedicated laboratories that test the side effects of new features, upgrades and configuration changes by profiling downscaled versions of their infrastructures in production against synthetic workloads.

In this section, the reutilization of the profiling data collected in the laboratory towards the validation of the queuing network model of the simulator is proposed. The downscaled version of the infrastructure used for validation is composed by a single datacenter, D, composed by four tiers T_{app}^{1x6}, T_{db}^{1x4}, T_{idx}^{1x4}, T_{fs}^{1x4}, where T_{db} and T_{fs} are connected to two san^{1x20} storage networks through 4 Gbps connections. T^{axb} indicates that the tier is composed by a servers with b cores each. Similarly, san^{axb} indicates that the storage network is composed by a servers and with b disks each. Tiers are connected through a 1 Gbps network. Profiling is carried out over the execution of a classic CAD application composed by eight operations: LOGIN, TEXT-SEARCH, EXPLORE, SPATIAL-SEARCH, FILTER RESULTS, SELECT, OPEN and SAVE. Each operation will have three variants: *light* (minimum resource requirement), *average* (average resource requirement) and *heavy* (maximum resource requirement). The (R_t, R_m, R_p, R_d) parameter set for each message for each operation variant is obtained by profiling the execution of each of these operations isolated within the laboratory infrastructure.

A *series* is defined as a serial sequence of these 8 operations pertaining to the same variant family launched by the one client every $\Delta\bar{t}$. The synthetic workload is generated by initiating a *light* series, *average* series and *heavy* series every $\Delta t(0)$, $\Delta t(1)$, $\Delta t(2)$ seconds respectively. Three experiments were carried out: $\Delta\bar{t} = \{10\ 24\ 40\}$, $\{12\ 29\ 48\}$, $\{15\ 36\ 60\}$. During these experiments multiple series overlap and messages pertaining

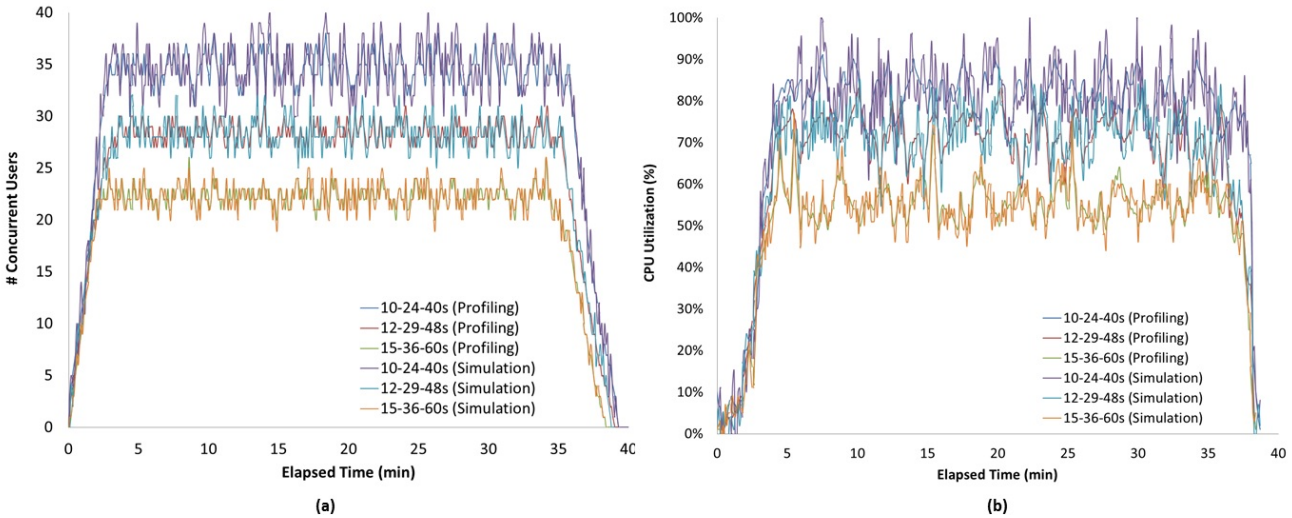

Figure 5. Validation Tests: # Concurrent users in D (a), CPU Utilization (%) in T_{app}^{1x6} (b)

Table I
MEAN PERCENTAGE ERROR (%) (MPE)

$\Delta\bar{t}$	$CPU\ T_{app}$	$CPU\ T_{db}$	$CPU\ T_{fs}$	$CPU\ T_{idx}$	$\#C$	R_t
$\{10\ 24\ 40\}$	9.52	11.07	5.66	7.13	6.03	6.30
$\{12\ 29\ 48\}$	9.44	11.23	4.97	6.75	4.79	6.77
$\{15\ 36\ 60\}$	8.67	9.87	5.72	7.42	5.63	4.81

to multiple operations flow concurrently through the queuing network model (figure 5 (a)). In this paper, we focused on the results for the validation of CPU utilization, along with response times and number of concurrent clients in the system. The results that compare the proposed model against the laboratory experiment are presented in figure 5 (b) and table I. Table I utilizes the Mean Percentage Error (MPE) metric to quantify the difference between simulation and profiling. The response time error, R_t, ranges between 4.81% and 6.30%, which is consistent with the results provided by previous research on analytic models and simulators [6] [12]. The number of concurrent users, $\#C$, shows an error between 4.79% and 6.03%. For CPU utilization, the minimum error observed was 4.97% for T_{fs} and the maximum error was 11.23% for T_{db}. The simulator predictions were pessimistic for all the experiments considered, resulting on an average CPU performance of the real system that was below the predicted values.

IV. CASE STUDY: GLOBAL DATA INFRASTRUCTURE OPTIMIZATION

In this section, we present a case study for a global IT infrastructure deployed in a Fortune 500 company. This organization aimed to cut infrastructure costs by reducing and consolidating the number of datacenters around the world. The simulator was used to analyze the impact of this adjustment on the client experience, while serving the same workload and background process requirements. The IT infrastructure runs three applications: 1) a Computer-Aided Design (CAD) software for engineers to design the product, 2) a Visualization (VIS) software that allows other departments of the company browsing parts of the product, and 3) a Product Data Management (PDM) software that facilitates the creation, management and publication of additional information about the product. These applications enable employees distributed across the world to work simultaneously sharing the same data files. Next, we show the simulation experiment that estimated a comparable service upon the reduction on the number of datacenters.

A. Topology

The simulated IT infrastructure is composed by six datacenters located in different continents: North America (D_{NA}), South America (D_{SA}), Europe (D_{EU}), Africa (D_{AFR}), Asia (D_{AS}) and Australia (D_{AUS}). D_{NA} is responsible for file synchronization between locations and indexing, in order to make sure that latest information is searchable and locally available for all the datacenters. Figure 6 illustrates the connectivity characteristics of the infrastructure, indicating bandwidth and latency between and within datacenters, along with the hardware specification of each tier.

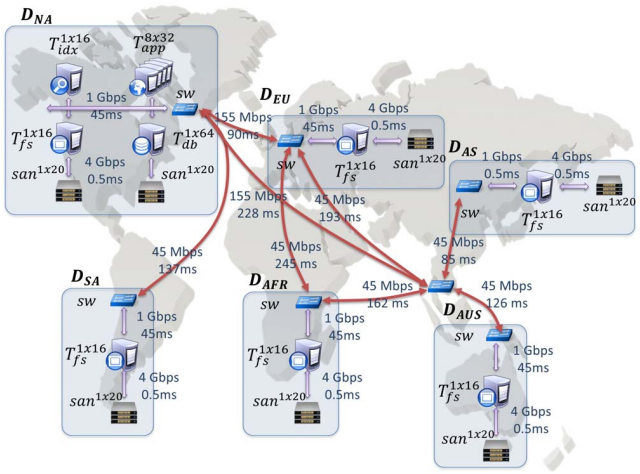

Figure 6. IT Infrastructure Topology and Tier specification

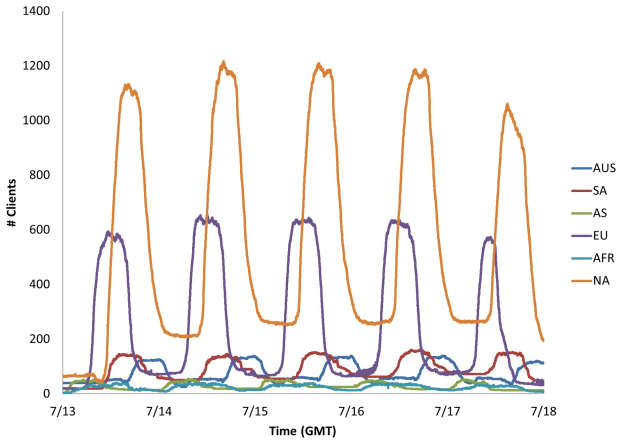

Figure 7. Workload variation through the day by datacenter

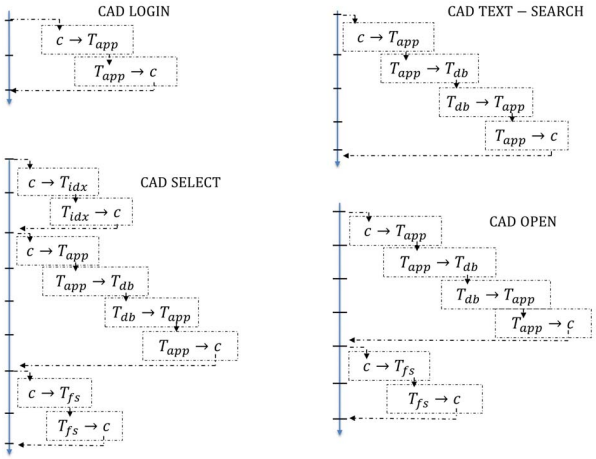

Figure 8. CAD Operation Trees (I)

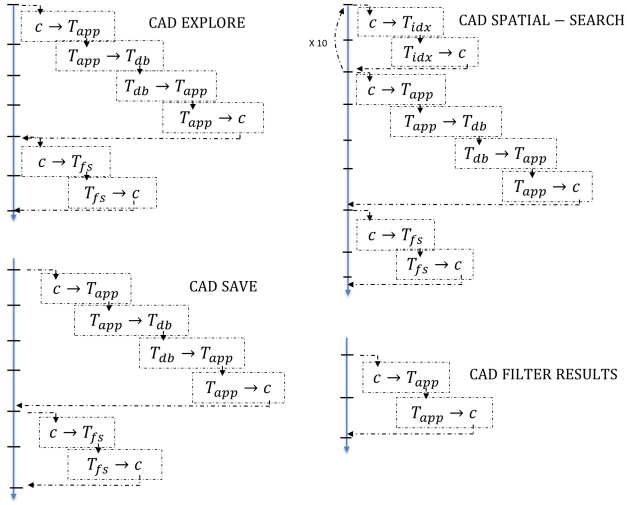

Figure 9. CAD Operation Trees (II)

B. Workload

Figure 7 illustrates the variation in the number of CAD clients using the system throughout the week for each datacenter. This information is provided as an input to the simulation along with the VIS and PDM workloads, which are omitted here for simplicity. In absence of information about the distribution of CAD operation types, for this study it is assumed to be uniformly distributed and to remain constant through time. Similar assumptions are made for VIS and PDM as well. Nevertheless, not all users in the system will be initiating operations every minute. For this use case, it is estimated that in a given minute, one third of the clients are *Active* from the total number of *Logged in* users. Active users initiate operations randomly distributed in that minute.

C. Operations

The simulator takes the message trees for CAD, VIS and PDM applications. Each message in the tree includes its parameter set (R_t, R_m, R_d, R_p). Parameter sets were obtained during the profiling process in the laboratory. Two operations may have identical message tree structure but differ on the parameter sets of one or many messages. Figures 8 and 9 illustrate the trees for the most common operations composing a CAD application. The trees for VIS and PDM follow analogous principles and are omitted here for simplicity.

D. Background Jobs

The simulated infrastructure has two background jobs: *Replication* and *Indexing*. These jobs run simultaneously with CAD, VIS and PDM client initiated operations, but as opposed to these, jobs are initiated by daemon processes

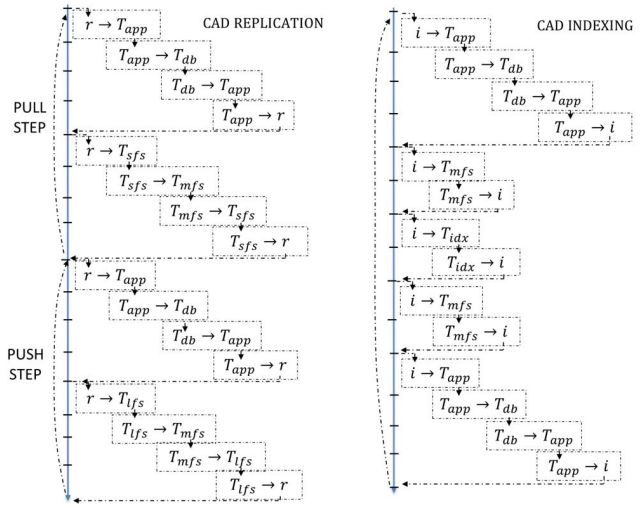

Figure 10. Background jobs: Replication and Indexing

running in D_{NA}. Both replication and indexing jobs are represented using the same message tree model, and just like in client operations, the (R_t, R_m, R_d, R_p) parameter set for each message was obtained in the laboratory.

1) Replication: Replication serves a double purpose: First, it creates multiple copies of each file increasing the reliability and fault-tolerance of the system. Second, it also distributes the latest version of each file across the world to allow clients downloading fresh data directly from their closest datacenter. Launching replication jobs with high frequency reduces the probability of a client requesting an outdated file, but requires more resources and can potentially degrade client experience. If a client requests an outdated file, the replication of this piece of data is triggered on demand from a remote datacenter, which takes considerably longer that the local datacenter download. In this infrastructure, replication follows a two step process: 1) *Pull*: For each slave datacenter, all modified files are copied to the master in batches. 2) *Push*: The master distributes copies of the modified files in batches to those datacenters with outdated versions. Not all slave datacenters need to receive copies during the *Push* step, filtering rules can be established if any slave datacenter is known not to use data from other datacenter. The simulation of the replication job requires to provide two types of data:

- *Data Growth*: It is necessary to measure the amount of new data in GB that is generated by each datacenter at each time of the day. Having a measure of the average file size, it is possible to feed to the simulator the number of new files generated during any interval of time.
- *Data Ownership*: The ownership distribution of the data manipulated in each site is used to determine the number copies that need to be pushed from the master datacenter to each slave.

The simulator takes as a parameter the frequency f_{rep} to launch the next replication job, and the batch sizes B_{pull} and B_{push}. In this case, $f_{rep} = 15$ min and $B_{pull} = B_{push} = 50$. If the duration of the replication job is longer than the frequency interval, multiple replication jobs will overlap. The simulator produces an estimation of the time it takes to synchronize datacenters at peak time, which reflects the maximum time interval during which the copy in slave datacenters can be outdated.

2) Spatial & Text Indexing: Indexing is critical in order to enable advanced features such as SEARCH. This infrastructure contemplates two types of indexing: *Spatial indexing* for 3D model search and *Text indexing* for text file search. The spatial indexing job produces a snapshot of the 3D model, which is used for 3D navigation and encapsulates spatial relationships between parts, whereas the text indexing job generates a word index that associates each word with the collection of documents that appears in. For simplicity, both types of indexes are constructed by T_{idx} within the master datacenter. Files copied to the master datacenter by the replication *Pull* step, along with files generated at the master datacenter itself, are queued in batches of B_{idx} files to be processed by the indexing job. Upon termination, the outcome of the indexing job for each file is registered at T_{db} and stored in T_{fs}. The simulator takes as a parameter the time gap G_{idx} to wait between indexing jobs. Hence, at any point in time only one indexing job is running. In this case $G_{idx} = 5$ min and $B_{idx} = 50$. The simulator produces an estimation of the time it takes to index files modified by all datacenters at peak time, which reflects the maximum time interval during which modifications made to a file will not be reflected in the search index and would be unsearchable.

E. Outputs

This section presents the outputs predicted by the simulator for a single day of the week.

1) Resource Utilization: Even though the simulator produces CPU, memory and network utilization estimates, in this paper we focused on the CPU utilization estimate, which was validated in section III. Figure 11 (a) illustrates the average CPU utilization by each tier in D_{NA}. For a T_{app}^{8x32} configuration, the expected average utilization in the peak time of the day was approximately 70%, while for T_{db}^{1x64}, T_{idx}^{1x16} and T_{fs}^{1x16} the CPU utilization ranges between 15-30% in the peak time of the day. The simulated T_{fs}^{1x16} tiers in slave datacenters produced similar results to the T_{fs}^{1x16} tier in the master datacenter.

2) Operation Response Time: The response times for CAD client operations in all datacenters was calculated. In particular the results for D_{NA} and D_{AUS} are presented in Figure 11 (b) and (d) respectively. The predicted values demonstrate that response times remain constant and are not degraded by the simultaneous execution of almost 6000

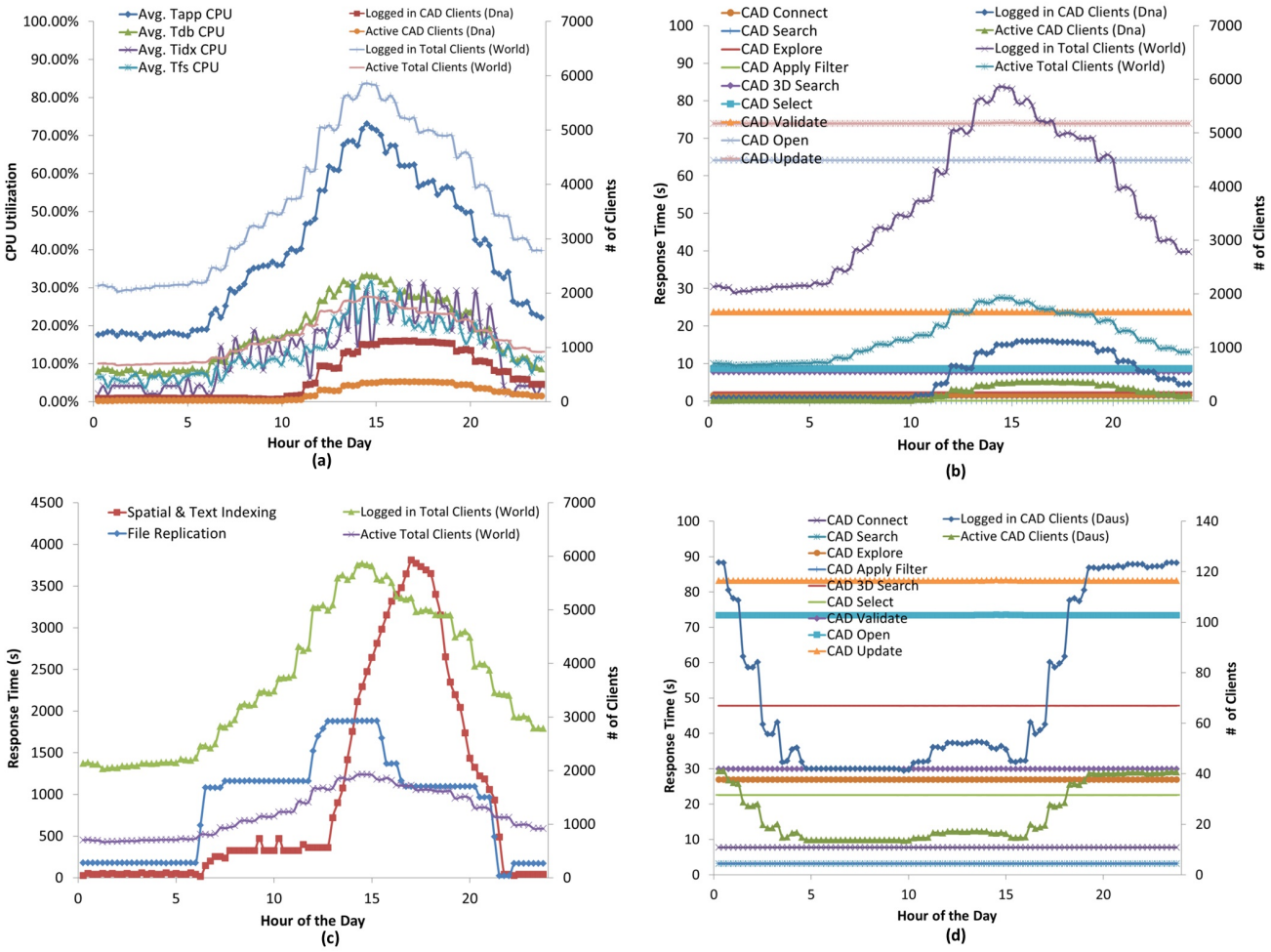

Figure 11. Outputs: CPU Utilization (a), CAD Response Times in D_{na} (b), Background Job Response Times (c), CAD Response Times in D_{aus} (d)

clients logged in (2000 active) at the peak, as long as servers run below CPU saturation limits. The response time difference between the same operation in D_{NA} and D_{AUS} is caused by an additional latency added to each individual message travelling from the slave to the master datacenter and viceversa.

3) Background Job Response Time: The estimation for the two background jobs running in D_{NA}, replication and indexing, is presented in Figure 11 (c). Under this insfrastructure configuration, the replication job is estimated to take 31 minutes to complete at the peak time of the day. This is the maximum time that a stale version of a file will be sitting on the T_{fs} of a slave datacenter. The indexing job is estimated to take 63 minutes. This is the maximum time that a file will be unsearchable for other clients in the system.

V. Conclusion

In this paper a simulator for global data infrastructures is presented. The simulator is based on a Multi-Agent System that imitates the behavior of an IT infrastructure composed by multiple datacenters running distributed software applications. For this purpose, two models were constructed: one for the hardware and another for the software. The behavior of the hardware components was reproduced using multiple queueing networks. The modularity of the design allowed interconnecting an arbitrary number of components, so as to build higher level elements such as server tiers and datacenters. Client-Server interactions initiated as part of a software application running on the infrastructure can be represented by trees of messages. Messages by multiple clients will flow concurrently through hardware components allocating resources and incurring on different processing costs. The simulation model was validated using data gathered from different applications running together on a downscaled version of the real infrastructure of a Fortune 500 company. The usefulness of the simulator is demonstrated by analyzing the case study of the same Fortune 500 company aiming to reduce costs by cutting down the number of datacenters, while keeping the same quality of service. As part of our

future work, we plan to investigate the use of statistical learning techniques to predict saturation of computational, storage or network resources, and if required, trigger fallback mechanisms to avoid possible downtime.

ACKNOWLEDGMENT

This work was supported by the EJ/GV Researcher Formation Fellowship BFI.08.80.

REFERENCES

[1] B. Gates, *Business at the Speed of Thought: Using a Digital Nervous System*. New York, NY, USA: Warner Books, Inc., 1998.

[2] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, pp. 41–50, January 2003.

[3] T. F. Abdelzaher, K. G. Shin, and N. Bhatti, "Performance guarantees for web server end-systems: A control-theoretical approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 13, pp. 80–96, January 2002.

[4] D. Villela, P. Pradhan, and D. Rubenstein, "Provisioning servers in the application tier for e-commerce systems," *ACM Trans. Internet Technol.*, vol. 7, p. 23, February 2007.

[5] M. Ahmad, A. Aboulnaga, S. Babu, and K. Munagala, "Modeling and exploiting query interactions in database systems," in *Proceeding of the 17th ACM conference on Information and knowledge management*, ser. CIKM '08. New York, NY, USA: ACM, 2008, pp. 183–192.

[6] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer, and A. Tantawi, "An analytical model for multi-tier internet services and its applications," in *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, ser. SIGMETRICS '05. New York, NY, USA: ACM, 2005, pp. 291–302.

[7] B. Urgaonkar, P. Shenoy, A. Chandra, P. Goyal, and T. Wood, "Agile dynamic provisioning of multi-tier internet applications," *ACM Trans. Auton. Adapt. Syst.*, vol. 3, pp. 1:1–1:39, March 2008.

[8] M. Yu, A. Greenberg, D. Maltz, J. Rexford, L. Yuan, S. Kandula, and C. Kim, "8th usenix symposium on networked systems design and implementation (nsdi 2011), march 30 - april 1, 2011, boston, massachusetts, usa, proceedings," in *NSDI*. USENIX, 2011.

[9] G. Ren, E. Tune, T. Moseley, Y. Shi, S. Rus, and R. Hundt, "Google-wide profiling: A continuous profiling infrastructure for data centers," *Micro, IEEE*, vol. 30, no. 4, pp. 65 –79, 2010.

[10] D. Gmach, Y. Chen, A. Shah, J. Rolia, C. Bash, T. Christian, and R. Sharma, "Profiling sustainability of data centers," in *Sustainable Systems and Technology (ISSST), 2010 IEEE International Symposium on*, May 2010, pp. 1 –6.

[11] J. Ellithorpe, Z. Tan, and R. Katz, "Internet-in-a-box: Emulating datacenter network architectures using fpgas," in *Design Automation Conference, 2009. DAC '09. 46th ACM/IEEE*, 2009, pp. 880 –883.

[12] S.-H. Lim, B. Sharma, G. Nam, E. K. Kim, and C. Das, "Mdcsim: A multi-tier data center simulation, platform," in *Cluster Computing and Workshops, 2009. CLUSTER '09. IEEE International Conference on*, 312009-sept.4 2009, pp. 1 –9.

[13] R. Buyya, R. Ranjan, and R. Calheiros, "Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities," in *High Performance Computing Simulation, 2009. HPCS '09. International Conference on*, 2009, pp. 1 –11.

[14] M. Niazi and A. Hussain, "Agent-based tools for modeling and simulation of self- organization in peer-to-peer, ad hoc, and other complex networks," *Comm. Mag.*, vol. 47, pp. 166–173, March 2009.

[15] S. Karnouskos and M. M. J. Tariq, "An agent-based simulation of soa-ready devices," in *Proceedings of the Tenth International Conference on Computer Modeling and Simulation*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 330–335.

[16] Y.-Q. Huang, H.-L. Li, X.-H. Xie, L. Qian, Z.-Y. Hao, F. Guo, and K. Zhang, "Archsim: a system-level parallel simulation platform for the architecture design of high performance computer," *J. Comput. Sci. Technol.*, vol. 24, pp. 901–912, September 2009.

[17] S. S. Lavenberg, *Computer Performance Modeling Handbook*. Orlando, FL, USA: Academic Press, Inc., 1983.

[18] E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik, *Quantitative system performance: computer system analysis using queueing network models*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1984.

[19] C. Shallahamer, *Forecasting Oracle Performance*. Berkely, CA, USA: Apress, 2007.

[20] K. K. Ramachandran and B. Sikdar, "A queuing model for evaluating the transfer latency of peer-to-peer systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, pp. 367–378, March 2010. [Online]. Available: http://dx.doi.org/10.1109/TPDS.2009.69

[21] A. Lebrecht, N. Dingle, and W. Knottenbelt, "Modelling and validation of response times in zoned raid," in *Modeling, Analysis and Simulation of Computers and Telecommunication Systems, 2008. MASCOTS 2008. IEEE International Symposium on*, sept. 2008, pp. 1 –10.

[22] A. S. Lebrecht, N. J. Dingle, and W. J. Knottenbelt, "Modelling zoned raid systems using fork-join queueing simulation," in *Proceedings of the 6th European Performance Engineering Workshop on Computer Performance Engineering*, ser. EPEW '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 16–29.

[23] E. Varki, A. Merchant, J. Xu, and X. Qiu, "Issues and challenges in the performance analysis of real disk arrays," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 15, no. 6, pp. 559 – 574, june 2004.

[24] Y.-L. Zhu, C.-Y. Wang, W.-Y. Xi, and F. Zhou, "Sansim - a platform for simulation and design of a storage area network." in *MSST'04*, 2004, pp. 373–384.