Modelling and Simulation of Load Balancing Strategies for Particle Physically Experiments at CERN

State of the Art of

Patrick Firnkes

December 1, 2017

at the Department of Informatics
Institute for Program Structures and Data Organization (IPD)

Reviewer: Jun.-Prof. Koziolek
Advisor: Jun.-Prof. Koziolek

Contents

1 Motivation

The *Worldwide LHC Computing Grid (WLCG)* is one of the largest grids in the world and processes the data created by the *Large Hadron Collider (LHC)* at CERN [6]. However, there is no model or simulation existing to evaluate the load balancing strategies of the grid.

The following chapter describes the WLCG in detail, the current state of the load balancing, and the desired state in the future.

1.1 Worldwide LHC Computing Grid

The WLCG processes 50 Petabytes of data each year, created by the LHC at CERN [25]. With its help the *Higgs boson* was discovered in 2012 [28].

The grid consists of 170 computing centres distributed in 42 countries. Combined it has 72.000 CPUs, 386 Petabytes online data space and 368 Petabytes nearline space (tape) [26]. Every day it runs 2 million jobs.

At CERN there are four experiments, namely: Atlas, Alice, CMS, LHCB. They all use the grid in a similar way, but not the same [5]. The WLCG is structured into three Tiers, which are described in 1.1. Tier 0 is the CERN compute centre, which stores all raw data of the experiments, makes the first pass reconstruction and distributes raw data to T1 compute centres. Tier 1 consists of 13 sites, which store a share of raw and reconstructed data. Tier 2 sites, which are mostly located at universities or other scientific institutes, do monte carlo production. In contrast to Tier 1 sites, the Tier 2 sites do not have much storage.

The WLCG is heterogeneous, meaning one site has different resources than an other site, the network connections between sites vary, and even the nodes in one site have different resources. It is trending to use more virtualization, resulting in even more heterogeneity [6].

Our work will focus on modelling and simulating load balancing strategies of the CMS experiment, but because the experiments use the grid in a similar way, the results of our work can be presumably transferred to the other experiments.

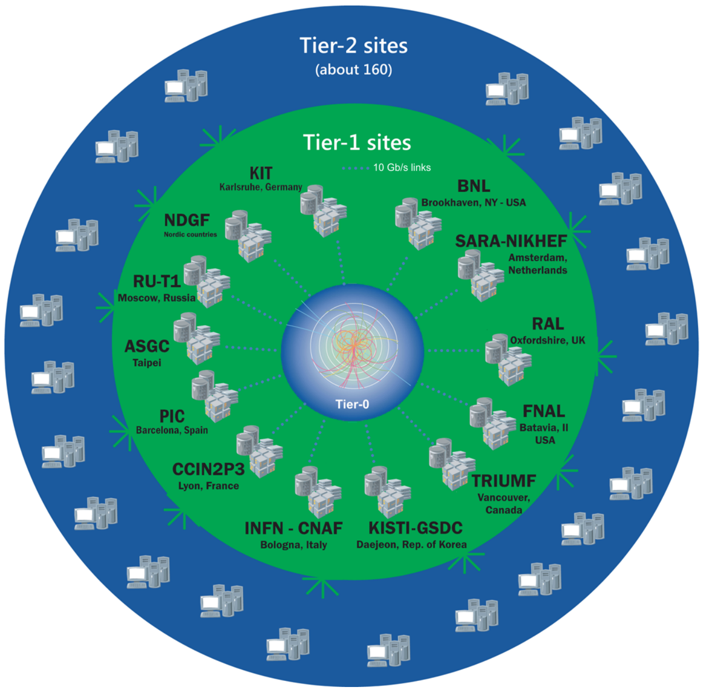


Figure 1.1: Tier structure of WLCG [27]

1.2 Current State

Currently the load balancing strategy of the WLCG is not optimal, e.g. it does not account the nature of the jobs. There are io-heavy jobs and compute-heavy jobs, if this nature is not accounted, it leads to a bad utilization of the nodes. It is desired, that the right amount of io-and compute-jobs are submitted to a site, so resulting in a good utilisation of IO and CPU.

HTCondor is the batchsystem, which is used for the WLCG and is the central point for the load balancing. It not only decides on which site the job should run, but even on which node, so it has full control over the load.

There are several other ideas how to improve the CMS computing model, but without a simulation no one can evaluate which one is best in terms of speed-up, cost or utilization.

1.3 Outlook

The amount of data produced by the LHC increases steadily, thus more computing resources are needed. However, these resources will not increase in the same magnitude as the produces data, thus better utilization of the given resources is required. Our approach will allow to simulate the effects of different load balancing strategies and decide on these results what the best strategy is. The scheduling problem itself is np complete, so the optimal solution will not be found, but we can decide on a set of strategies which one is the best [24].

Further, our approach can be used to evaluate the effects of changing the grid. Thus, we can decide what the best approach is to improve the CMS computing model, e.g. adding cache, create a new network connection or allow dynamic nodes. This is hard to decide without simulation, because one cannot consider all side-effects, e.g. increasing the CPU resources may be useless, because then the network could become the bottleneck.

2 Foundations

In this chapter *Palladio* is described, which is intended to be used as the tool to model and simulate the CMS computing model. An other important foundation is to explain what the differences between grids and clouds are.

2.1 Palladio

Palladio is a software architecture simulator developed by Karlsruhe Institute of Technology (KIT), FZI Research Center for Information Technology, and University of Paderborn. The development started 2003 and it is still actively developed today. Palladio predicts quality of software properties (e.g. performance) using several models of a system [3].

These models are the component model, the assembly model, the resource model, the allocation model, and the usage model. The component model specifies the structure and behaviour of the components independently from their later usage. This allows reuse of the components, but requires its parametrization. The assembly model specifies how the components are connected to model the software architecture. The resource model describes the resources environment. The allocation model describes how the different components are mapped to the resources. Finally, the usage model describes how the system is used [3].

After these models are created, Palladio can be used to simulate the system, choosing one of several supported simulators. This process is shown in 2.1.

Palladio was extended with the *architectual templates* to make it possible to model cloud computing environments efficiently [20].

SimuLizar is another simulator for Palladio to simulate self-adaptive systems [2]. It was later extended to support the metrics scalability, elasticity, and efficiency [20].

The first case studies using Palladio in cloud environments show that the results are accuracy [20]. However, Palladio was never used to model and simulate such a large system as the CMS computing model.

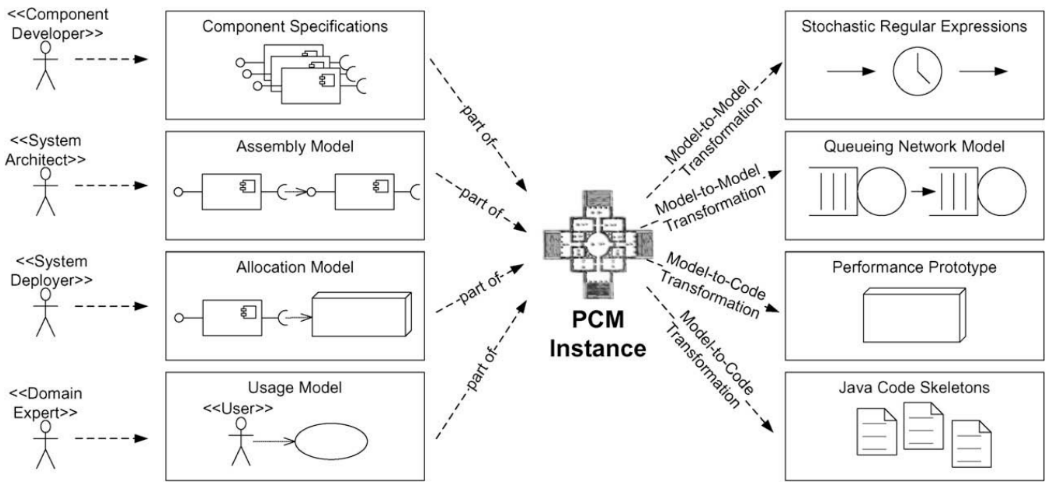


Figure 2.1: Process of Palladio [3]

2.2 Differences Grid-Computing and Cloud-Computing

Grid-computing are distributed systems working together to solve a problem. Cloud-computing has a lot in common with grid-computing, because it is also a distributed system working together and because it evolved from grid-computing [13]. However, there are also some important differences.

Some of the key aspects are: virtualization, on-demand resource provisioning, and elasticity [13]. Firstly, cloud-computing uses massively virtualization, this means that assigning application models to computing nodes do not model the abstraction correctly TODO [8]. Secondly, the cloud allows on-demand resource provisioning [13]. Further, clouds are elastic, they scale up or down according to the currently required resources and allow autoscaling of applications, meaning the provisioned resources to an application are dynamically changed [8].

3 State of the Art

There are a lot other approaches beside Palladio that deal with the simulation of grids or clouds. In this chapter a selection of these is presented.

3.1 Grid Simulators

Grid simulators exist since the late 1990s, the most ones have been created in the mid 2000s. However, most simulators are not developed any more. In this section the most common ones are presented.

3.1.1 SimGrid

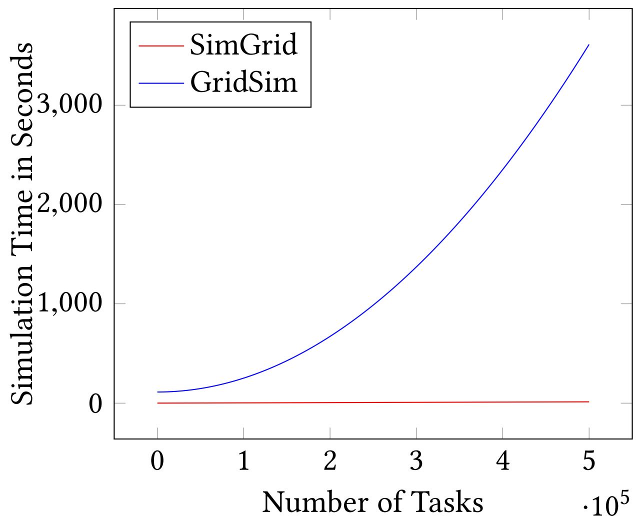
SimGrid is a framework for large-scale distributed systems and one of the earliest created grid simulators [9]. Its first release was 1998 and is one of the few that is still developed today. Further, its one of the most used simulators [9].

In SimGrid one has to define the platforms, and the Deployment in XML and the application and scheduling in C. It supports simulation of CPU, I/O, and network resources, heterogeneous workloads and heterogeneous platforms [9].

It is more scalable than GridSim, which is an other popular grid simulator [9]. 3.1.1 shows the simulation time of SimGrid and GridSim for 2000 nodes and increasing tasks. SimGrid performs significantly better that GridSim, because GridSims simulation time increases quadratic with the number of tasks, while SimGrids only increases linear. At 500.000 tasks GridSim requires more than one hour and 4.4 GiB of memory to simulate, while SimGrid needs less than 14 seconds and with only 165 MiB.

This performance difference is created by a lot of optimizations, including implementing light-weight execution contexts, using lazy activity updates, and using trace integration for resource management [9].

Simulation Times of SimGrid and GridSim [9]



3.1.2 GridSim

GridSim was created 2001 as part of the gridbus project and was developed until 2010 [7]. It is focused on evaluating scheduling and resource brokering [1]. It supports simulation of CPU, I/O and network resources, heterogeneous workloads and heterogeneous platforms [7]. Its implemented in Java, using SimJavas discrete event simulation [7]. It is popular and has been used from other members of the community.

However, as in section 3.1.1 mentioned, it does not scale very well and further it does not deliver realistic results regarding moving or the storage of data [24] .

3.1.3 GangSim

GangSim derived from an enhancement of Ganglia monitoring toolkit in 2005 [11]. It was not further developed after its initial release. It is used to handle usage constraints for sites and virtual organizations. It evaluates allocation of individual resource from the interactions between allocation policies across virtual organisations [11].

It supports simulation of CPU, I/O and network resources, heterogeneous workloads and heterogeneous platforms [11].

In difference to most other simulators, it is a discrete time-stepped simulators [17]. This means it will simulate every time-step, even if nothing happens. This results in a bad performance and bad scalability.

3.1.4 OptorSim

OptorSim was created 2003 as part of the European Data Grid project and is used to evaluate data replication strategies in grids [4]. To do that it simulates the time needed to copy or access the required files for each job and replicating files according to a given strategy.

It is not developed any more and does not consider job scheduling optimization. Because of its little scope, it does not support simulating of CPU and I/O resources, and heterogeneous platforms [4].

3.1.5 DGSim

To improve the simulation environments the DGSim framework focuses on automating and optimizing the overall simulation process [17]. It helps setting up the experiment by to generating realistically grids, generate grid dynamics and evolution events, and generate realistic grid workload or use traces of existing ones. Further, it enables grouped or batches of simulations [17].

It was created in 2007 at Delft University of Technology, and is not public available, so nobody else used it.

3.1.6 ChicagoSim

The data locality is a important aspect in grid computing, thus ChicagoSim was created in 2003 to evaluate differnt scheduling and data replication algorithms [29]. In contrast to OptorSim it also evaluates scheduling algorithmns. One of the results is that scheduling jobs to sites containing the data and replicating popular data-sets is a good solution and works better than scheduling jobs to idle processors and moving the data.

ChicagoSim was not further developed, it does not support simulating CPU or I/O, instead simple constant waiting time is used, without any concurrence to resources [29].

3.2 Cloud Simulators

In the recent years the research went from grid simulators to cloud simulators, because grid simulators cannot sufficiently model the cloud infrastructure [21]. Although the WLCG is a grid and therefore a grid simulator would be more suitable that an cloud simulator, it is useful to know about the latest developments in cloud simulators. Therefore this sections presents a brief overiew about the most common simulators.

3.2.1 CloudSim

CloudSim is the most used and most sophisticated cloud simulator [21]. Created 2010 at the University of Melbourne as part of the cloudbus project. It is based on GridSim and itself is the foundation for a lot of other simulators. It shows a good scalability, simulation 100.000 machines in less than 5 min, requiring only 75 MB of RAM [30]. It supports both system and behaviour modelling of cloud system components such as data centers, virtual machines and resource provisioning policies.

It was later extended with a network simulation component to evaluate applications which communicate [14].

3.2.2 CDOSim

CDOSim is simulator based in CloudSim with the purpose to evaluate competing cloud deployment options (CDOs) [12]. It was developed 2012 at Kiel University. It represents more user than provider perspective and hide fine fine-grained internals of a cloud platform. It helps to find best ratio between high performance and low costs, by helping to select a cloud provider, the mapping between service and virtual machines and adaption strategies.

3.2.3 LocalitySim

Another simulator based on CloudSim is LocalitySim, which was created 2016 at the University of Cairo [16]. Data locality affects the performance of any scheduling algorithm. If the scheduler places the job far away from its data, extra time is needed to transfer data. Most other cloud simulators are not supporting data locality.

3.2.4 GreenCloud

GreenCloud focuses on the energy efficiency of clouds [18]. It was created 2012 at the University of Luxembourg. It focuses specifically on the measurement of energy consumption of severs, switches and links. It simulates the network on packet level, because of this it network model is very accurate.

However, it assumes that every server only has a single core and virtualization, storage area networks and resource management are not considered [30].

3.2.5 Cloud Autoscaling Simulation

Researches at the Czech Technical University tried to evaluate predictive autoscaler, but found that CloudSim had accuracy and speed issues [32]. Thus, they implemented a new simulation method based on queueing models, delivering realistic results.

They found out that the autscaling is very stable and cost-efficient when they scale up based on latency and scale down based on utilization.

3.3 The MONARC Project

The MONARC project was started 1998 to help with the inital design of the WLCG at CERN [10]. They created models and simulations of possible structures of the grid. For that the MONARC simulator was developed. The results indicated that a hierarchical structure with regional centers would perform best, thus we have the 3 Tier structure today [22].

2006 a second version of the MONARC simulator was released, which is not only usable for the WLCG but general large scale distributed computing systems [19]. But one clearly see that its origin is WLCG. In difference to its first version a discrete-event approach instead of a discrete time-stepped was used.

2010 Zach et al. [34] used MONARC2 to simulate the ALICE Tier-2 site in Prague. The simulation included 3000 CPUs, and the CPU, network and I/O resources were simulated.

The results are accurate, however they could not simulate the network load correctly and MONARC2 already ran into performance issues, making it not usable for even larger systems.

3.4 CACTOS

CACTOS is a approach to cloud infrastructure automation and optimization [23]. It includes the CACTOS Runtime Toolkit for monitoring and resource management and the CACTOS Prediction Toolkit for evaluation of alternative data center deployment scenarios, and resource management algorithms. The predication toolkit is built on Palladio and SimuLizar. The models of the predication toolkit regarding resources or application behaviour is updated using monitoring data, this is done to handle the highly dynamic behaviour of clouds. F. Pop et al. [24] suggested the idea to couple monitoring and grid simulation tools in 2006.

CACTOS supports the rapid testing of resource management algorithms due the ability to simulate them without reimplementing them for the Prediction Toolkit and automatic creation of workload models by using runtime monitoring data.[31].

First case studies show that the predication toolkit has a high accuracy, but it was not tested on a large system yet [31].

4 Discussion

Several grid and cloud simulators have been presented in chapter 3. Grid simulators are a better option for modelling and simulating the CMS computing model, see section ??] for the differences of grid and cloud computing. It may be possible to use a cloud simulator, but the added functionality may give an overhead and to model a grid in cloud may be needed workaround and the result would not be as easy to understand. For cloud simulators CloudSim is the most evolved one.

Table 4 compares the grid simulators based on our needs. As one can see only SimGrid and Palladio do full fill them, but Palladio its not clear if its as scalable as needed. As part of the CACTOS project Palladio and SimuLizar are made ready to use for cloud, thus we expect that it will scale. All other simulators are either not developed any more, have never been intended to be used as a general simulater, and have only be created to research one small aspect. Most simulators support the simulation of CPU, I/O and network ressources. Also heterougnous platforms and tasks are supported.

As mentioned in 3 OptorSim, ChicagoSim and LocalitySim are the only simulators which consider data locality. However, this feature is not needed for simulating load balancing strategies, because we only need to effects of a data replication strategy, e.g. reduce data miss rate 30%. The data locality feature is needed if one wants to evaluate a better data replication strategies.

There was no indication that a similar big grid or cloud as the CMS computing grid was simulated yet, so our work will evaluate if the existing tools make it possible to do it.

Table 4.1: Comparison of Grid Simulators

Simulator	Active Project	General usable	Scheduling	CPU	I/O	Network	Heterogeneous Platforms	Heterogeneous Tasks	Scalable
SimGrid	✓	✓	✓	✓	✓	✓	✓	✓	✓
GridSim	✗	✓	✓	✓	✓	✓	✓	✓	✗
GangSim	✗	✗	✓	✓	✓	✓	✓	✓	✗
OptorSim	✗	✗	✗	✗	✗	✓	✗	✓	?
DGSim	✗	✗	✓	✗	✓	✓	✓	✓	✓
ChicagoSim	✗	✗	✓	✗	✗	✓	✓	✓	✓
MONARC	✗	✓	✓	✓	✓	✓	✓	✓	✗
Palladio	✓	✓	✓	✓	✓	✓	✓	✓	?

5 Conclusion

- reasonable to use Palladio, case study was good
- transform models into SimGrid
- others outdated
- no try to model and simulate similar big grid

6 Outlook

- Use Palladio to model , simulate and validate Tier 1 Center GridKA
- After that use these results to model whole Grid

Bibliography

[1] Hussein Al-Bahadili. *Simulation in Computer Network Design and Modeling: Use and Analysis: Use and Analysis.* IGI Global, 2012.

[2] Matthias Becker, Steffen Becker, and Joachim Meyer. "SimuLizar: Design-Time Modeling and Performance Analysis of Self-Adaptive Systems." In: *Software Engineering* 213 (2013), pp. 71–84.

[3] Steffen Becker, Heiko Koziolek, and Ralf Reussner. "The Palladio component model for model-driven performance prediction". In: *Journal of Systems and Software* 82.1 (2009). Special Issue: Software Performance - Modeling and Analysis, pp. 3–22. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2008.03.066. URL: http://www.sciencedirect.com/science/article/pii/S0164121208001015.

[4] William H. Bell et al. "Optorsim: A Grid Simulator for Studying Dynamic Data Replication Strategies". In: *The International Journal of High Performance Computing Applications* 17.4 (2003), pp. 403–416. DOI: 10.1177/10943420030174005. eprint: https://doi.org/10.1177/10943420030174005. URL: https://doi.org/10.1177/10943420030174005.

[5] Ian Bird. "Computing for the Large Hadron Collider". In: *Annual Review of Nuclear and Particle Science* 61.1 (2011), pp. 99–118. DOI: 10.1146/annurev-nucl-102010-130059. eprint: https://doi.org/10.1146/annurev-nucl-102010-130059. URL: https://doi.org/10.1146/annurev-nucl-102010-130059.

[6] Ian Bird et al. *Update of the Computing Models of the WLCG and the LHC Experiments.* Tech. rep. 2014.

[7] Rajkumar Buyya and Manzur Murshed. "GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing". In: *Concurrency and Computation: Practice and Experience* 14.13-15 (2002), pp. 1175–1220. ISSN: 1532-0634. DOI: 10.1002/cpe.710. URL: http://dx.doi.org/10.1002/cpe.710.

[8] Rodrigo N Calheiros et al. "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms". In: *Software: Practice and experience* 41.1 (2011), pp. 23–50.

[9] Henri Casanova et al. "Versatile, Scalable, and Accurate Simulation of Distributed Applications and Platforms". In: *Journal of Parallel and Distributed Computing* 74.10 (June 2014), pp. 2899–2917. URL: http://hal.inria.fr/hal-01017319.

[10] Monarc Collaboration et al. *Models of Networked Analysis at Regional Centres for LHC experiments: Phase 2 report.* Tech. rep. Technical Report CERN/LCB-001, CERN, 2000. http://www.cern.ch/MONARC, 2000.

[11] Catalin L Dumitrescu and Ian Foster. “GangSim: a simulator for grid scheduling studies”. In: *Cluster Computing and the Grid, 2005. CCGrid 2005. IEEE International Symposium on*. Vol. 2. IEEE. 2005, pp. 1151–1158.

[12] Florian Fittkau, Sören Frey, and Wilhelm Hasselbring. “CDOSim: Simulating cloud deployment options for software migration support”. In: *Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2012 IEEE 6th International Workshop on the*. IEEE. 2012, pp. 37–46.

[13] Ian Foster et al. “Cloud computing and grid computing 360-degree compared”. In: *Grid Computing Environments Workshop, 2008. GCE’08*. Ieee. 2008, pp. 1–10.

[14] S. K. Garg and R. Buyya. “NetworkCloudSim: Modelling Parallel Applications in Cloud Simulations”. In: *2011 Fourth IEEE International Conference on Utility and Cloud Computing*. Dec. 2011, pp. 105–113. DOI: `10.1109/UCC.2011.24`.

[15] Mrs R Kingsy Grace, S Swathi Priya, and S Surya. “A survey on grid simulators”. In: *Int. J. Comput. Sci. Inf. Technol. Secur* 2.6 (2012), pp. 1224–1230.

[16] Ahmed Hassan Abase, Mohamed Khafagy, and Fatma Omara. “Locality Sim : Cloud Simulator with Data Locality”. In: 6 (Dec. 2016), pp. 17–31.

[17] Alexandru Iosup, Ozan Sonmez, and Dick Epema. “DGSim: Comparing grid resource management architectures through trace-based simulation”. In: *European Conference on Parallel Processing*. Springer. 2008, pp. 13–25.

[18] Dzmitry Kliazovich, Pascal Bouvry, and Samee Ullah Khan. “GreenCloud: a packet-level simulator of energy-aware cloud computing data centers”. In: *The Journal of Supercomputing* 62.3 (2012), pp. 1263–1283.

[19] Iosif C Legrand et al. “Monarc simulation framework”. In: *International Workshop on Advanced Computing and Analysis Techniques in Physics Research, Tsukuba, Japan*. 2003.

[20] Sebastian Lehrig and Matthias Becker. “Approaching the cloud: Using palladio for scalability, elasticity, and efficiency analyses”. In: *Proceedings of the Symposium on Software Performance*. 2014, pp. 26–28.

[21] Rahul Malhotra and Prince Jain. “Study and comparison of various cloud simulators available in the cloud computing”. In: *International Journal* 3.9 (2013).

[22] Youhei Morita, Monarc Collaboration, et al. “Validation of the MONARC simulation tools”. In: *Computer Physics Communications* 140.1-2 (2001), pp. 153–161.

[23] P. O. Östberg et al. “The CACTOS Vision of Context-Aware Cloud Topology Optimization and Simulation”. In: *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*. Dec. 2014, pp. 26–31. DOI: `10.1109/CloudCom.2014.62`.

[24] F. Pop et al. “A Simulation Model for Grid Scheduling Analysis and Optimization”. In: *International Symposium on Parallel Computing in Electrical Engineering (PARELEC’06)*. Sept. 2006, pp. 133–138. DOI: `10.1109/PARELEC.2006.8`.

[25] María Alandes Pradillo et al. “Optimising costs in WLCG operations”. In: *Journal of Physics: Conference Series* 664.3 (2015), p. 032025. URL: `http://stacks.iop.org/1742-6596/664/i=3/a=032025`.

[26] WLCG Project. *WLCG REsource, Balance & USage*. 2017. URL: `https://wlcg-rebus.cern.ch/apps/capacities/federations/` (visited on 11/2017).

[27] WLCG Project. *WLCG Tier Centers*. URL: `http://wlcg-public.web.cern.ch/` (visited on 11/2017).

[28] WLCG Project. *WLCG Worldwide LHC Computing Grid*. 2017. URL: `http://wlcg-public.web.cern.ch/` (visited on 11/2017).

[29] Kavitha Ranganathan and Ian Foster. "Simulation Studies of Computation and Data Scheduling Algorithms for Data Grids". In: *Journal of Grid Computing* 1.1 (Mar. 2003), pp. 53–62. ISSN: 1572-9184. DOI: `10.1023/A:1024035627870`. URL: `https://doi.org/10.1023/A:1024035627870`.

[30] Georgia Sakellari and George Loukas. "A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing". In: *Simulation Modelling Practice and Theory* 39.Supplement C (2013). S.I.Energy efficiency in grids and clouds, pp. 92–103. ISSN: 1569-190X. DOI: `https://doi.org/10.10/j.simpat.2013.04.002`. URL: `http://www.sciencedirect.com/science/article/pii/S1569190X13000658`.

[31] Christian Stier et al. "Rapid Testing of IaaS Resource Management Algorithms via Cloud Middleware Simulation". In: *ACM / SPEC International Conference on Performance Engineering (ICPE'18)*. ICPE. 2018.

[32] T Vondra and J Šedivy. "Cloud autoscaling simulation based on queueing network model". In: *Simulation Modelling Practice and Theory* 70 (2017), pp. 83–100.

[33] Y Wu et al. *Simulating the Farm Production System Using the MONARC Simulation Tool*. 2001.

[34] Č Zach et al. "Simulation of the job processing performance at an ALICE Tier-2 site with MONARC". In: *Journal of Physics: Conference Series* 331.7 (2011), p. 072038. URL: `http://stacks.iop.org/1742-6596/331/i=7/a=072038`.