Int j simul model 15 (2016) 1, 70-82
ISSN 1726-4529 Original scientific paper

SIMULATION-BASED PERFORMANCE ANALYSIS OF THE ALICE MASS STORAGE SYSTEM

Vickovic, L.[*,**]; Gotovac, S.[**] & Celar, S.[**]
[*] European Organization for Nuclear Research – CERN, CH – 1211 Geneva 23, Switzerland
[**] Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture – FESB, University of Split, R. Boskovica 32, 21 000 Split, Croatia
E-Mail: Linda.Vickovic@cern.ch, Sven.Gotovac@fesb.hr, Stipe.Celar@fesb.hr

Abstract
CERN – the European Organization for Nuclear Research today, in the era of big data, is one of the biggest data generators in the world. Especially interesting is transient data storage system in the ALICE experiment. With the goal to optimize its performance this paper discusses a dynamic, discrete event simulation model of disk based Storage Area Network (SAN) and its usage for the performance analyses. Storage system model is based on modular, bottom up approach and the differences between measured and simulated values vary between 1.5 % and 4 % depending on the simulated component. Once finished, simulation model was used for detailed performance analyses. Among other findings it showed that system performances can be seriously affected if the array stripe size is larger than the size of cache on individual disks in the array, which so far has been completely ignored in the literature.
(Received in April 2015, accepted in October 2015. This paper was with the authors 1 month for 1 revision.)

Key Words: Big Data, Mass Storage System Optimization, Storage Area Network Simulation, Storage Area Network Optimization, Hierarchical Performance Modelling and Analysis

1. INTRODUCTION

With the goal to study even further in the structure of the matter, at CERN, the European Organization for Nuclear Research, the Large Hadron Collider (LHC) has been constructed. ALICE (A Large Ion Collider Experiment) is one of the five general purpose experiments hosted by the LHC with the aim to study the physics of strongly interacting matter at extreme energy densities. During data acquisition, ALICE produces a dataflow of 15 GB/s that needs further processing and filtering by the Data Acquisition System (DAQ) and High Level Trigger (HLT). As a result of that processing, the bandwidth of the outgoing dataflow from the DAQ is reduced to 1.25 GB/s for archiving in the Mass Storage System (MSS). Such throughput produces more than 1 PB of data every year. The whole data flow is presented in Fig. 1.

According to [1] a big data is a term that defines data with three main characteristics. First, it involves a great volume of data. Second, the data cannot be structured into regular database tables and third, the data is produced with great velocity and must be captured and processed rapidly. As all of these three characteristics can be applied to the ALICE mass storage system, it can be placed it in the category of 'big data'.

For physicists, the biggest issue is how to search and find interesting event in such a huge amount of data, but the scope of this article is to determine how to provide optimal storage capacity that will satisfy 1.25 GB/s input to the storage system.

In order to satisfy the requirements for storage capacity and input bandwidth, the MSS is organized on two levels. The first one is responsible for the real time data acquisition and is called the Transient Data Storage (TDS), and the second, the so called Permanent Data Storage (PDS) should provide enough storage capacity for all data acquired during the lifetime of the experiment. The TDS implementation presents a particular challenge, as it has

to satisfy online requirements for data storage of 1.25 GB/s and enough capacity to store all data before they are transferred in PDS at slower speed. In practice, the ALICE TDS system is organized as a bunch of disk arrays organized in a Storage Area Network (SAN).

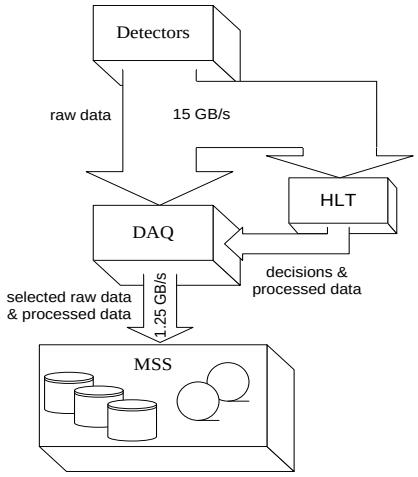

Figure 1: The ALICE data flow.

In order to perform capacity planning and predict the performance of the required ALICE transient data storage one of possible approaches is to develop a system model. Although model development is a time-consuming process, once it is developed it can be used for different analyses. Also, optimal system configuration can be chosen by simply changing the system components, their organization or input parameters without any additional expenses.

The first part of this paper briefly describes the development of a discrete event simulation SAN model in Ptolemy. Although this whole section could contain a lot more details, the model development process is already published in [2-5]. Here only the basic idea used for model development is presented, while the main contribution of the paper is elaborated in the second part.

In the second part the paper shows how a developed model could be used to explore the performance of the ALICE transient data storage under various sets of circumstances and as a result how to optimize it. Although the described model has been used for the optimization of the ALICE mass storage system, all described methods and models are general enough and could be used to predict the behaviour of any SAN system based on disk drives and disk arrays, with large, sequential workload.

The structure of the paper is organized as follows. Section 2 gives the survey of related work, while Section 3 describes the main characteristics of the ALICE workload and the architecture of the system used for model verification. Storage system simulation model development process is presented in Section 4. Section 5 gives detailed SAN system performance analyses based on the developed model. Finally, conclusions and plans for future work are presented in Section 6.

2. SURVEY OF THE RELATED WORK

Today, with the huge amount of data generated from many sources, an efficient storage system is more necessary than ever. Although the focus is now mainly set on cloud computers, the storage elements are mostly disk based and they are still under study as described in [6]. The authors investigate benchmarking and modelling for a disk-based storage system in order to design and build a practical storage tier.

The study of SAN has gained popularity as this technology has emerged as one of the most commonly deployed storage solutions for the massive amount of data. As mentioned in

previous section, if all necessary hardware is available, the easiest way to predict system behaviour and performance is an experimental evaluation. Regarding SAN such an approach is described in [7]. Although the authors have carried out experiments in the context of a specific system, it is suggested that their methodology permits storage-system designers to evaluate empirically their system performance with considerable confidence.

Regardless of their suggestions, numerous analytical and simulation models were developed during years. For example, some analytical models of SAN are focused on individual parts of storage network like performance modelling of disks [8], or data access model [9]. In [10] the authors used M/G/1 queue model to model a single server connected to SAN with the goal to model multiple complex database applications running concurrently. Other analytical models explore the behaviour of the whole storage system. For example [11] proposed the use of an M/M/Q model for performance simulation of SAN. However, none of these models is capable of describing the performance of a multi-level RAID architecture. In [12] a queuing model is developed that approximates the effect of synchronizations that can be used to model adaptive multi-level RAID systems in which the RAID level appropriate to an application is selected dynamically. Further on, in [13] the focus from storage devices is transferred to optimization and heuristic methods for the design of medium to large scale SANs.

While all these analytical models are usually best suited for the investigation of steady-state behaviour the simulation ones capture the dynamic nature of real-life workloads [14]. As a result numerous studies have examined discrete event simulation of SAN.

One of the first was a discrete event simulator "SimLab", discussed in [15]. It was implemented to aid the development and verification of distributed algorithms for real-time delivery of data in storage network. After that an event driven simulator, called SANSim was presented in [16]. It was aimed exclusively for FC arbitrated loop (FC-AL) based SANs. Further on, [17] presented a discrete event simulator based on OPNET to model the synchronous write operations during I/O traffic and in [18] a FC based switched fabric SAN simulator is described. It is also based on OPNET with initial focus to simulate the FC-2 level of the FC protocol dealing with fabric login process on a single FC switch. In [19] a simulation framework for SAN management, named iSAN (imitation Storage Area Network), is proposed. It can be used to perform a simulation of management module ranging from individual device to large scale multi-vendor heterogeneous SAN for enterprise and for what-if-analysis of enterprise IT environment before modelling the changes. SANBlaze, described in [20], has attempted emulation of storage devices (disk drive, tape drive or arrays) by manufacturing target and initiator emulators for Fiber Channel, SAS and other storage protocols.

Most models referred above address monitoring, tuning and autonomic management of storage subsystems, while not taking into account the interoperability with other tiers (e.g., databases) and its impact on performance. Work discussed in [14] provides an environment for the performance evaluation of various architectures, and supports experimentation with different configuration policies and with a variety of workloads.

Also, in [21] a discrete event simulation models for two SAN designs are presented with the aim to compare the performance of the two designs, and also to evaluate the performance of the second design under various component-failure scenarios. This paper, however, is theoretical – a complete simulation is yet to be produced.

The work presented in this paper actually follows the work already presented in [3-5]. It is a part of the last group, a group of dynamic, discrete event simulators and it gives a simple disk based SAN simulator that can be used to evaluate how different parts of system or different workloads can influence overall performances.

3. WORKLOAD SPECIFICATION AND SYSTEM ARCHITECTURE FOR MODEL VERIFICATION

Workload specification in this simulation is defined by the requirements of the ALICE TDS system that receives only two types of requests: for data storage from the DAQ and for data transfer from MSS. Such requirements imposed relatively simple workload definition for TDS simulation: large (> 300 MB) sequential exclusively write or read requests. It is especially important for disk array simulation because it allowed considering only full stripe writes, and thereby simplifying the simulation.

The actual architecture of the TDS system is shown in the left part of Fig. 2. It consists of several servers, switch and array based SAN. For test measurements that architecture was simplified and all servers were replaced by just one, which produces both: write requests like DAQ and read requests like PDS. A SAN that forms the TDS was also replaced with one single disk and arrays used for verification of single disk and array simulation results respectively. The architecture of that system, used for model verification, is presented in the right part of Fig. 2.

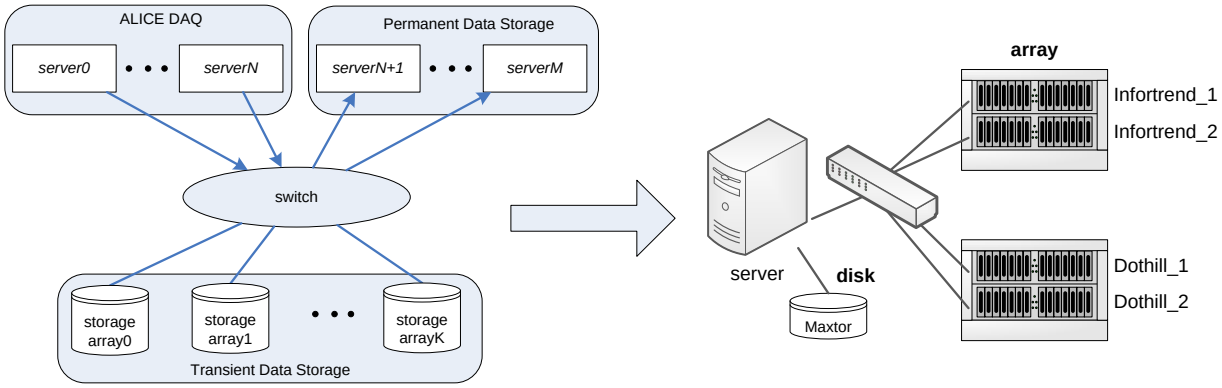

Figure 2: Digital engineering for productivity improvement.

To verify simulation results following storage media was used:

- Maxtor MaXLine Plus II disk;
- two identical Infortrend arrays (IFT-7250F) with 5 Ultra ATA Maxtor MaXLine Plus II disks;
- two identical DotHill arrays (SANnet II 200 FC Array) with 5 Cheetah X15 36LP FC disks;
- server(s) that generate write or read workload.

Disk and arrays used for the measurements were dedicated just for the tests, without any operating system installed. As a result, the whole data transfer from and to the storage media was produced only by the performance measurements program. The program is a standalone client running in the background and it constantly writes or reads predefined fixed length records (8, 32, 128, 512, 2048, 8192, 32768, 131072 kB) filled with random data until the predefined output file size (2 GB) is reached. In the same time, it monitors the exact duration of each transfer. As the writing to disk is not a constant process (faster on disk's outer tracks) in each measurement 15 files were written (each 2 GB) and then average write/read time was counted.

So, the first step was to perform a series of measurements. For each file size all record sizes were tested and required time was stored in the file. Then, the whole thing was repeated on developed simulator and once more time of simulation was stored. So, the difference between these two was used for model validation.

Although hardware used for test measurements may seem a bit obsolete, it does not have any impact on simulation model and its appliance as far as the same technology is used. All relevant data about disks and arrays are defined in configuration file and can be easily changed if simulation requires.

4. DISCRETE EVENT SIMULATION MODEL OF SAN

This section describes model development process for dynamic, discrete event (DE) simulation model of SAN as well as results of model verification. The model itself was created in Ptolemy.

Hierarchical decomposition described in [22] was used to produce this model of SAN and it is combined with the bottom up approach. Its advantage is in the model granularity that allows easily changing configuration of storage system under study. As a result, in this model at the beginning the focus is set on the lowest storage component, disk drive. When the disk drive model is developed and verified, the focus is expanded from disk to disk array, and then finally to array based SAN.

Disk drive model development model is given in more details in [2], while disk array is discussed in [4] and in [3] where some interesting facts regarding parity cache placement within model are discussed.

Although both models should be used only for simulation of large sequential write or read streams, they mimic the behaviour of real storage elements good enough to be used as a basis for further SAN simulation. The difference between measured and simulated results is presented in Table I and it is lower than any published in literature till now. For example, for single disk the lowest published difference is in [24] and it is about 5 %, while for array is in average 9 % in [25] and 15 % in [23]. The detailed discussions on how these results have been obtained are already presented and can be found in [2-4], as mentioned before.

Table I: Difference between measured and simulated results.

	Difference [%]		
	Single disk	Ultra ATA Array	FC-AL Array
Write workload	0.98	1.5	3.16
Read workload	0.8	2.5	2.8

4.1 SAN model

Model development of a SAN is the final step in the production of the ALICE TDS system simulation. Due to modular approach, a fully functional model of SAN requires just an interconnection between already developed components – a model of a switch. So, the scheme presented in Fig. 3 is used as a basis for a SAN model development, a switch that connects servers with storage media.

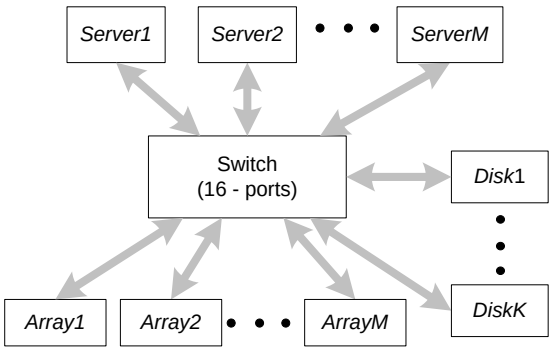

Figure 3: The general scheme of a SAN model.

A model of a non-blocking switch described in [26] is used as a basis for switch module architecture and it is presented in Fig. 4. According to that, a switch model should have several inputs and outputs, each with its own queue and each input queue should have direct connection to every output queue.

Once again, the specific architecture of ALICE TDS system simplified initial model. According to [27] before the data transfer starts, each storage element is assigned to specific server and there is no switching until the data transfer ends. As the connections between servers and storage elements are specified before the beginning of the simulation and they are constant during the simulation run, the switch model can be simplified with direct connections as shown in Fig. 5.

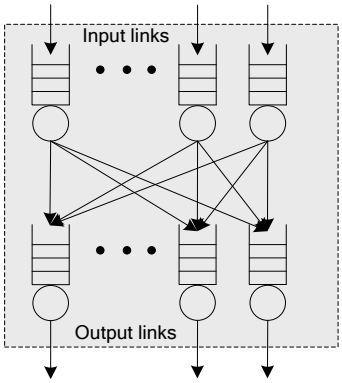

Figure 4: The general scheme of a switch module.

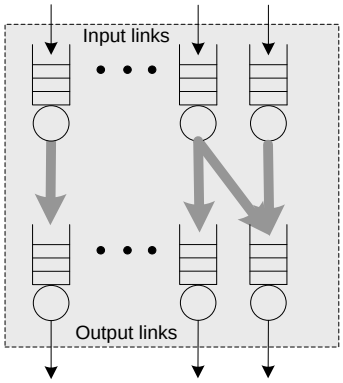

Figure 5: The custom scheme of a switch module.

Further on, according to its specifications, even the influence of switching on Fibre Channel (FC), non-blocking switch to the resulting system performances can be completely ignored. As a result, in the final SAN model a Switch module is used just to establish the connection between servers and storage elements.

However this model describes just the physical architecture of a switch, while a storage virtualization engine should be modelled separately. A model of the storage virtualization engine would introduce a new component – a masking engine that should hide the real physical architecture of storage elements from servers. Similar logic is applied as a part of dacCache module described in [4].

In the ALICE TDS system each disk array in the storage pool is divided in two or three virtual storage containers, so the virtual storage area has two or three times more individual storage containers then the physical ones. But according to the specific design of the ALICE TDS system just one virtual container from the physical array is active in a given moment, and has the full use of all resources (cache, bandwidth, etc.) on that array. In this way just the capacity of each virtual container is reduced while all other characteristics are the same like on the real disk array. As a result, modelling of virtualization engine is simplified and it is incorporated in the model.

With the aim to test this model, disk and arrays were connected to server through non-blocking switch and all measurements were performed once more. The results measured and simulated for this specific model of a SAN are the same ones produced for individual array or disk measurements and simulations. Such results approved initial assumptions that the delay produced by a switch can be completely ignored and the resulting architecture can be treated as a bunch of the individual servers communicating with the individual storage elements.

5. SIMULATION-BASED ANALYSES OF SAN PERFORMANCE

There are several issues that should be discussed with the goal of optimizing SAN based storage system performance. Thus, this section first explores the impact of the workload type

on the system performance and after that it asses the data organization within arrays in the SAN.

All presented throughputs are the results of the simulation executed on a model presented in Fig. 6. Although four servers are presented in the model, majority of simulations uses just one server. Multiple servers are used just to test system behaviour under multiple streams. Also, each simulation was run on two different types of arrays, Infortrend Ultra ATA and Dothill FC one.

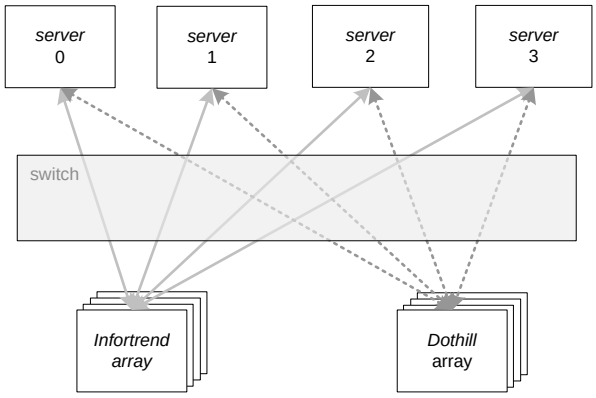

Figure 6: The architecture model of the simulated system.

5.1 Workload tuning

The developed model was used to test several workload parameters. First, and the most obvious one, is the selection of the optimal file size. However, in the case of ALICE DAQ the size of one event is 32 MB and that fact actually eliminates all the other record sizes from further considerations because events splitting or merging would generate additional data. Further on, a bunch of 32 MB events are grouped into files and a bigger file means less data to run catalogue database [27], so the maximal available file size (2 GB) is going to be used.

The next step in the workload tuning process is to explore how multiple, simultaneous write or read streams on the same storage array can influence the system performances. The simulation showed that the average bandwidth (a bandwidth achieved by each server participating in the communication) for multiple streams is reversely proportional to the number of streams. It means that if more servers participate in communication each achieves lower bandwidth. As a result it is possible to conclude that multiple write or read streams from multiple servers toward the same storage element should be avoided because they will not provide sufficient bandwidth for servers.

The third and the final step in the workload tuning is to explore how simultaneous read and write streams on the same storage element influence its performances. In this case simulation showed that the produced bandwidth for write stream in a simultaneous workload is just a little lower than a single write stream, and it can be used. The problem emerges for read stream. The produced bandwidth is even lower than an average bandwidth for two simultaneous read streams. Such downhill of read performances for simultaneous read and write streams suggests that simultaneous reading and writing from the same storage element should not be used under any circumstances.

5.2 Data layout optimization

The main building element of the ALICE TDS is the disk array and as a result, the data layout optimization is related with the choice of the array level and the appropriate stripe unit size. As the ALICE TDS employs a RAID5 array configuration this section explores the influence of stripe unit size on system behaviour.

The stripe unit determines how a logical request from a user is broken up into physical requests for storage on the individual disks in the array. The problem with the size of the stripe unit is actually the problem of the optimal placement of data on disks in the array with the goal to balance the load across the disks, minimize the response time of individual requests and maximize the throughput of the system [28]. Actually, a large stripe unit will tend to keep a file clustered together on a few disks (possibly one), while a small stripe unit tends to spread each file across many disks. A RAID5 stripe size is in general regarded as the larger the better. However, the real answer depends on the access pattern – large or small, sequential or random, read or write, etc. Actually, many researchers' studies were conducted to discover how the stripe size influences the array performances and for example in [29] it is given a model to count a stripe size based on the system load. Other studies, described in [30] and [26] even produced an equation that gives a heuristic approximation of the optimal stripe unit for the synchronized, block striped arrays.

To explore how the performance of the ALICE TDS can be improved by the change of stripe size, different stripe sizes were tested on a developed simulation model. At the beginning, the simulated bandwidth is simply observed as a function of the record and the stripe size. The results are presented in Figs. 7 and 8. For read workload small stripe sizes (8 and 32 kB) produced the lowest bandwidth and the linear growth of stripe size caused logarithmic growth of the performances for all record sizes.

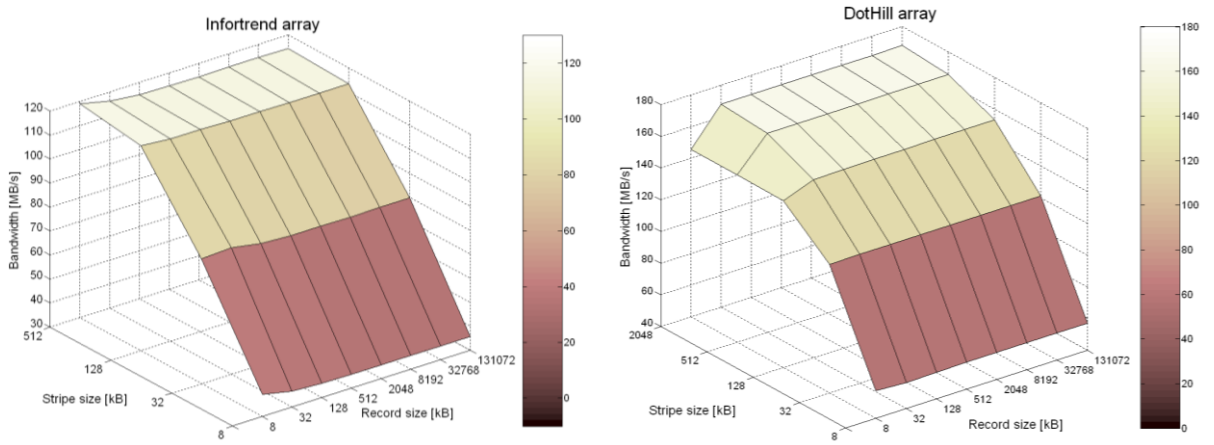

Figure 7: The read performance for Infortrend and DotHill array for different stripe and record sizes.

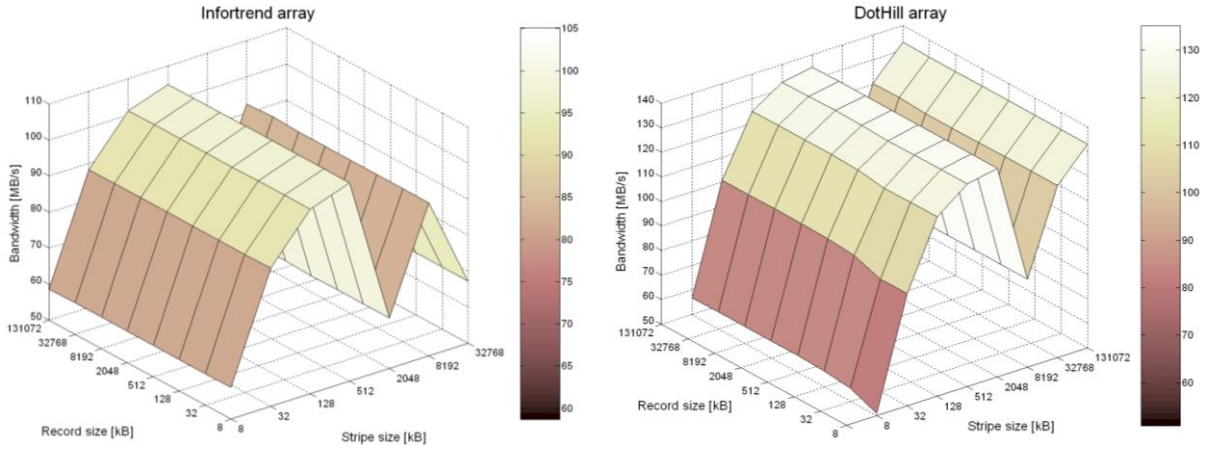

Figure 8: The write performance for Infortrend and DotHill array for different stripe and record sizes.

The produced bandwidth for write workload is a bit more complex. In the beginning it grows logarithmically with the stripe size but at one point, there is a huge performance downfall. Further increase of stripe size improves the performance, but the bandwidth never

reaches the highest value. The performance dropping shoved to be related to the size of disk cache on individual disks in the array. In the case of Infortrend array, disk cache is 2 MB and performance downfall happened for 2046 kB stripe size. For DotHill, 8196 kB stripe caused performance dropping and its disk cache is 8 MB. Further increase of stripe size resulted with the bandwidth growth, but internally all requests are split to match the size of disk cache. So, to avoid additional latencies caused by stripe splitting, internally on disks, stripe size should be smaller than a disk cache and in the rest of the section only smaller stripes are going to be analysed.

From this example it is possible to conclude that the system performances are definitely sensitive on the change of stripe size, and the stripe size should be as large as possible, but still smaller than the size of cache on individual disks that form array.

The idea for the next step in the data layout optimization process emerges from [30] and [31] where the number of concurrent user requests at one time is discussed. As a result Figs. 9 and 10 show the influence of stripe size and the number of concurrent streams on system performance.

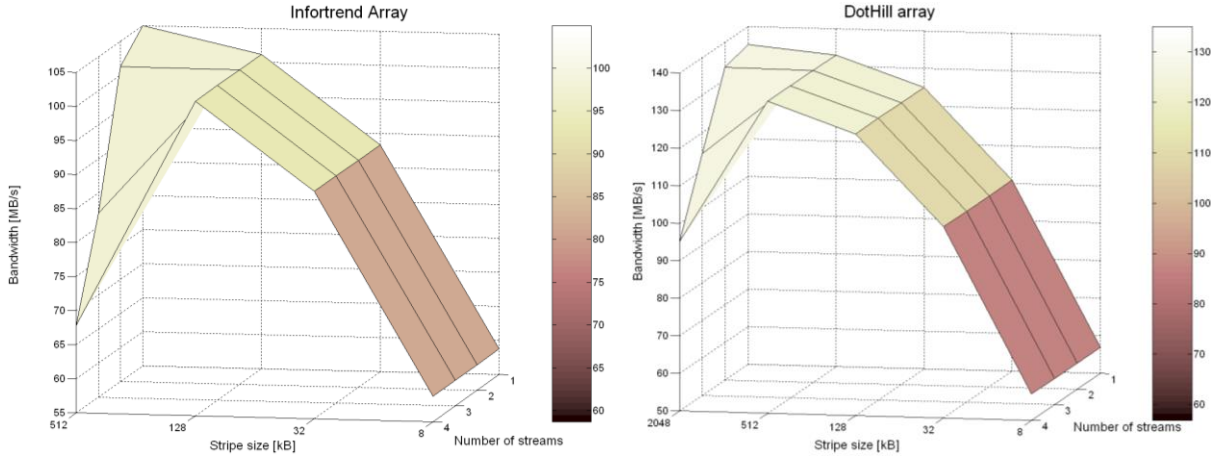

Figure 9: The influence of simultaneous streams and different stripe sizes for write workload.

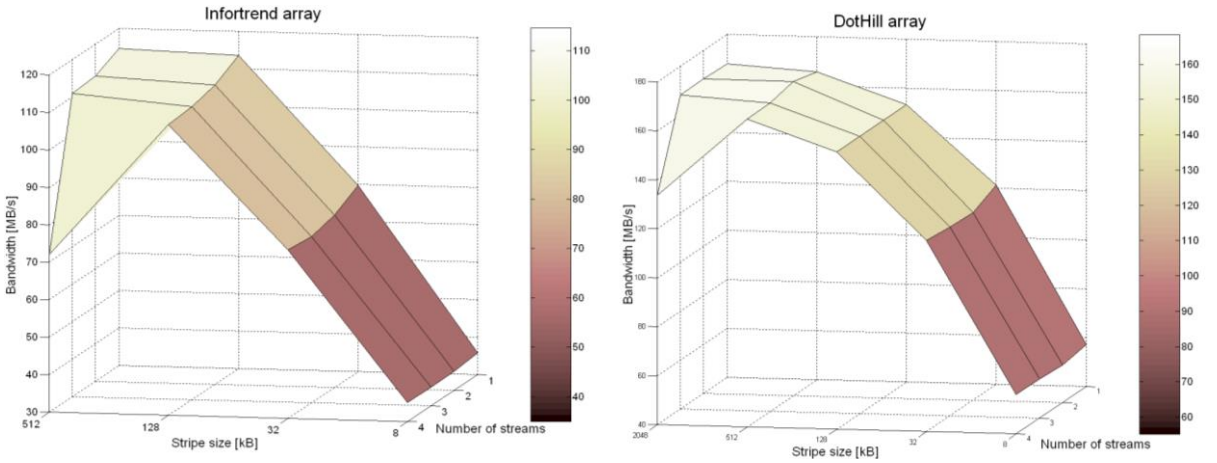

Figure 10: The influence of simultaneous streams and different stripe sizes for read workload.

Like in previous case the bandwidth grows logarithmically with the stripe size, while the number of concurrent streams just barely influences the system performance. The deflection happens only for four simultaneous streams and the highest record size. With four concurrent streams, each trying to write a certain stripe size (512 kB for Infortrend and 2048 kB for DotHill) on the same array, a disk cache is again a bottleneck because the cache size is exceeded. This conclusion also imposes additional restraint on the optimal stripe size

selection and it is not compatible with [30] and [31] where optimal stripe size should be proportional with the number of concurrent streams without any limitations. So, in the case of large sequential workload a stripe size is inversely proportional to the number of concurrent streams.

The final idea for system optimization, related with the stripe size, refers to the connection between the stripe size and the number of disks in the array. Actually, researches described in [31] concluded that for write workload optimal stripe size should grow proportionally with the number of disks. The simulation results presented in Fig. 11 acknowledge that statement, but they also showed that to achieve the same bandwidth, arrays with more disks need smaller stripe size. Finally, the best performances for write workload are achieved for the largest stripe size, with the most disks in the array.

According to [31] for a read workload, the optimal stripe unit should vary inversely to the number of disks. The results produced for a read workload simulation, presented in Fig. 12 showed that the biggest difference in the achieved bandwidth is between 3 and 5 disks in the array, while further increasing the number did not influence the performance significantly. On the contrary to [31], a developed model proved to be insensitive to the number of disks in the array and the best results are produced for the highest stripe size.

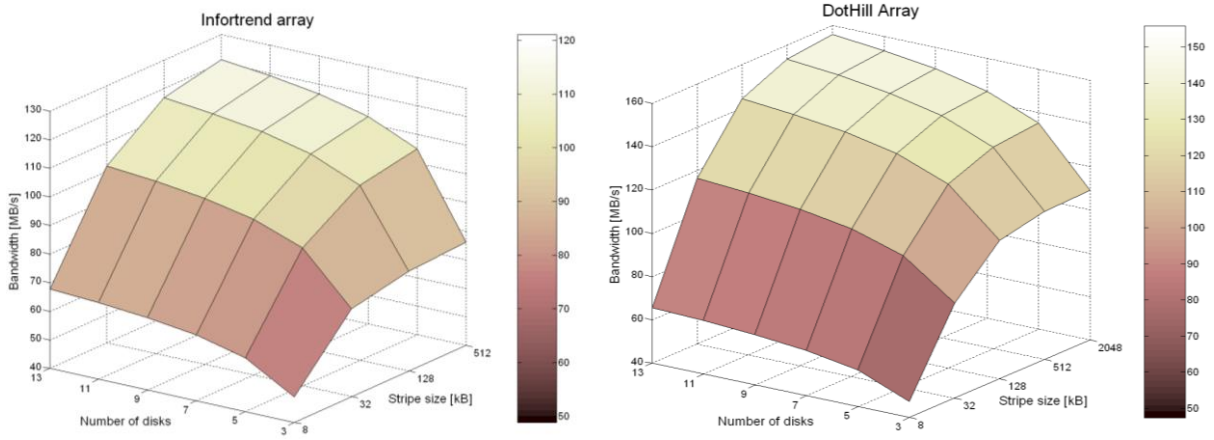

Figure 11: The influence of disk number in the array and stripe sizes on write performances for Infortrend and DotHill array.

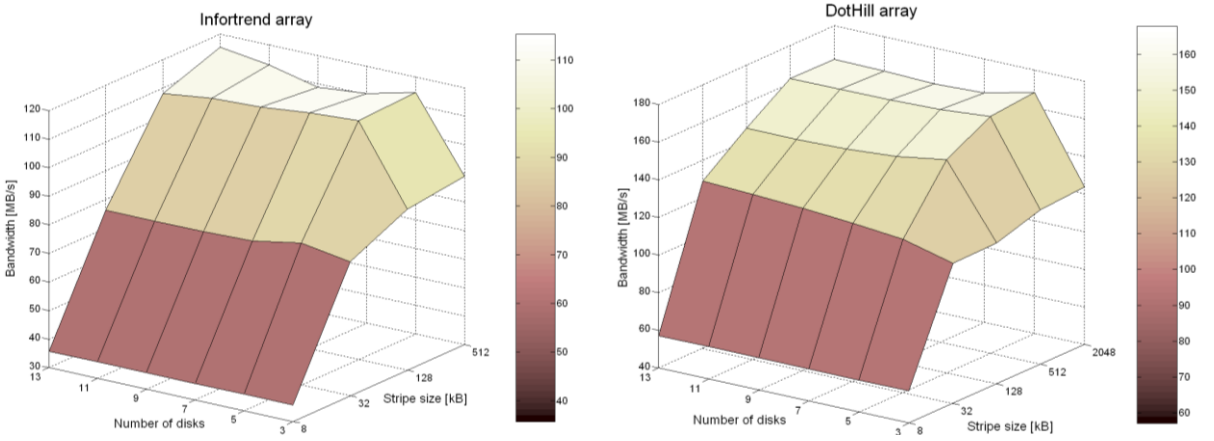

Figure 12: The influence of disk number in the array and stripe sizes on read performances for Infortrend and DotHill array.

The results published in [31] and the ones produced by the model presented in this paper are not similar. Nevertheless, it is not totally unexpected because some elementary assumptions differ in the model development process for those models. All models presented

in the literature till now have completely ignored the size of disk cache [8] and in this research the size of disk cache is proved to be a critical factor. Also, the models described in literature were intended to explore different write policies (read modify write, reconstruct write and full stripe), while the model elaborated in this paper assumed large, sequential write requests that result with full stripe writes, and other write policies are not considered.

From all discussions and figures presented in this section it is possible to conclude that the influence of the stripe size cannot be ignored. The general rule for just one read or write stream should be: larger stripe size will produce better performances. But the stripe size should not exceed the cache size of individual disks in the array. Resulting bandwidth with several concurrent streams is unchanged by the number of streams, until again the number of concurrent requests on each disk reaches the disk cache size. So, if the systems with more concurrent streams are going to be used, choosing the smaller stripe sizes should be wiser because the performance loss in the case of the smaller stripe size is less pronounced than in the case when the total number of concurrent streams exceeds the disk cache size. Finally, the number of disks also influences the performances, especially for write workload, but the growth in the bandwidth form 5 to 13 disks is about 3 % for read and 13 % for write workload, while the system costs are increased significantly.

6. CONCLUSIONS

Mass storage systems, mostly in the form of Storage area network (SAN), are widely used in different areas today. With the goal of understanding and improving their performance, different models are developed and explored. The work presented in this paper could be divided in two major parts. The first one presents the development of a dynamic, discrete event simulation model of SAN, while the second part uses that model to perform different performance analyses for the Transient Data Storage (TDS) system under workload defined by the ALICE experiment.

Hierarchical decomposition, used for the model development in this case was combined with the bottom up approach. As a result, the focus was firstly set on the lowest storage component, disk drive. When the disk drive model had been developed and verified, the focus was moved to disk array, and then finally to array based SAN. The difference between simulated throughput and throughput measured on available testing system is summarized in Table I. However, such results are not without limitation. The discussed model is customized for the requirements of the ALICE TDS, which means it is going to be used for a storage system simulation with sequential reading and writing of large files. With the given limitation, the developed model is still general enough to be used for simulation of any SAN environment based on disks and disk arrays (FC and ultra ATA) organized in the RAID 5 mode. Also, it is simple enough to be easily extended to support future devices by modelling new, more complicated, components and workloads than those which are validated. As it is mentioned before, the whole model development process is presented in small details, as that is already published in [2-5].

The second part of the paper uses SAN's simulation to explore the impact of workload and data layout parameters on overall storage system performances.

The workload analyses were focused on the influence of the file and record size on system performances, and they showed that the best solution for record size is 32 MB (the size of one ALICE event). Further on with the goal to minimize number of entries in run catalogue database [27], maximal available file size (2 GB) is chosen as optimal.

With the defined file and record size the next step was to analyse the workload pattern from servers to storage subsystems. That analysis leads to the decision that the optimal system performance will be achieved only if one server is involved in data transfer with single

storage array. The server in that communication can generate single or multiple streams of the same kind (read or write) which will not degrade system performances. Streams from multiple servers toward single storage array should be avoided, because they lower the achieved bandwidth on each server and can lead to insufficient bandwidth toward transient data storage system. Simultaneous reading and writing of the same array should also not be used because they lead to the huge degradation of read performances.

Further on, the simulation shoved that the system performances are sensitive to the change of stripe size in the array. The stripe size should be as large as possible, but still smaller than the size of cache on individual disks that form the array. That is also the main novelty presented in this paper, as the influence of disk cache on performances of array based storage system was completely ignored in literature till now.

In the future, it would be interesting to introduce a virtualization engine at the SAN level. The virtualization engine is already included as a part of disk array but its implementation on SAN level will enable additional analyses of SAN architecture related to the effect of storage element virtual organization on the system performances.

ACKNOWLEDGEMENT

Thanks to all people from the ALICE DAQ Lab, who helped us with measurements, and especially thanks to Pierre Vande Vyvre.

REFERENCES

[1] Garlasu, D.; Sandulescu, V.; Halcu, I.; Neculoiu, G.; Grigoriu, O.; Marinescu, M.; Marinescu, V. (2013). A big data implementation based on Grid computing, *Proceedings of the 11th Roedunet International Conference (RoEduNet)*, 1-4

[2] Vickovic, L.; Celar, S.; Mudnic, E. (2010). Disk drive simulation model development, Katalinic, B. (Ed.), *DAAAM International Scientific Book 2010*, DAAM International, Vienna, 535-548

[3] Vickovic, L.; Mudnic, E.; Gotovac, S. (2009). Parity information placement in the disk array model, *COMPEL – The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, Vol. 28, No. 6, 1428-1441, doi:10.1108/03321640910991994

[4] Vickovic, L.; Celar, S.; Mudnic, E. (2011). Disk array simulation model development, *International Journal of Simulation Modelling*, Vol. 10, No. 1, 27-37, doi:10.2507/IJSIMM10(1)3.174

[5] Vickovic, L. (2007). *Management and optimization of mass data storage system for ALICE experiment*, PhD thesis, University of Split, Split

[6] Lee, D.-J.; O'Sullivan, M.; Walker, C. (2012). Benchmarking and modeling disk-based storage tiers for practical storage design, *ACM SIGMETRICS Performance Evaluation Review*, Vol. 40, No. 2, 113-118, doi:10.1145/2381056.2381080

[7] Bokhari, S.; Rutt, B.; Wyckoff, P.; Buerger, P. (2006). Experimental analysis of a mass storage system, *Concurrency and Computation: Practice and Experience*, Vol. 18, No. 15, 1929-1950, doi:10.1002/cpe.1038

[8] Zeng, G.-P.; Chlamtac, I.; Zhu, H. (2003). Performance modeling of disks in Storage Area Networks, *Proceedings of the 1st International Workshop for SAN*, Dallas

[9] Singh, A.; Voruganti, K.; Gopisetty, S.; Pease, D.; Duyanovich, L.; Liu, L. (2005). A hybrid access model for Storage Area Networks, *Proceedings of the 22nd IEEE/13th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST'05)*, 181-188

[10] Arzuaga, E.; Kaeli, D. R. (2008). An M/G/1 queue model for multiple applications on Storage Area Networks, *Proceedings of the 11th Workshop on Computer Architecture Evaluation Using Commercial Workloads (CAECW 2008)*, 25-32

[11] Li, C.; Zhou, L.-Z.; Xing, C.-X. (2005). Using MMQ model for performance simulation of Storage Area Network, *ICDE 2005: Proceedings of the 21st International Conference on Data Engineering*, 1269, doi:10.1109/ICDE.2005.301

[12] Harrison, P.; Zertal, S. (2007). Queuing models of RAID systems with maxima of waiting times, *Performance Evaluation*, Vol. 64, No. 7-8, 664-689, doi:10.1016/j.peva.2006.11.002
[13] Walker, C. G.; O'Sullivan, M. J. (2010). Core-Edge design of storage area networks – A Single edge formulation with problem-specific cuts, *Computers & Operations Research*, Vol. 37, No. 5, 916-926, doi:10.1016/j.cor.2009.07.005
[14] Aizikowitz, N.; Glikson, A.; Landau, A.; Mendelson, B.; Sandbank, T. (2005). Component-based performance modeling of a Storage Area Network, *Proceedings of the 2005 Winter Simulation Conference*, 2417-2426
[15] Berenbrink, P.; Brinkmann, A.; Scheideler, C. (2001). SIMLAB – a simulation environment for Storage Area Networks, *Proceedings of the 9th IEEE Euromicro Workshop on Parallel and Distributed Processing*, 227-234
[16] Wang, C.-Y.; Zhou, F.; Zhu, Y.-L.; Chong, C. T.; Hou, B.; Xi, W.-Y. (2003). Simulation of fibre channel Storage Area Network using SANSim, *ICON 2003 – The 11th IEEE International Conference on Networks*, 349-354
[17] Rueda, A.; Pawlak, M. (2003). Performance modeling on synchronous write operations of Storage Wide Area Network (SWAN), *Proceedings of the OPNETWORK 2003*, 6 pages
[18] Staley, J.; Muknahallipatna, S.; Johnson, H. (2007). Fibre channel based Storage Area Network modeling using OPNET for large fabric simulations: Preliminary work, *32nd IEEE Conference on Local Computer Networks*, 234-236
[19] Routray, R.; Gopisetty, S.; Galgali, P.; Modi, A.; Nadgowda, S. (2007). iSAN: Storage Area Network management modeling simulation, *International Conference on Networking, Architecture, and Storage*, 199-208
[20] Sanblaze Technology, Inc., SANBlaze storage emulation, from *http://www.sanblaze.com/storage-emulation*, accessed on 15-07-2013
[21] Walker, C. G.; O'Sullivan, M. J.; Elangasinghe, M. (2005). *Evaluation of Core-Edge Storage Area Network Designs using Simulation*, School of Engineering Technical Report No. 627, School of Engineering, University of Auckland, 24 pages
[22] Ammar, R. A. (2007). Hierarchical Performance Modeling and Analysis of Distributed Software Systems, Rajasekaran, S.; Reif, J. (Eds.), *Handbook of Parallel Computing: Models, Algorithms and Applications*, Chapman & Hall/CRC Press, Boca Raton, 122-144
[23] Uysal, M.; Alvarez, G.; Merchant, A. (2001). A modular, analytical throughput model for modern disk arrays, *Proceedings of the 9th International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems*, 183-192
[24] Ruemmler, C.; Wilkes, J. (1994). An introduction to disk drive modeling, *Computer*, Vol. 27, No. 3, 17-28, doi:10.1109/2.268881
[25] Varki, E.; Merchant, A.; Xu, J.; Qui, X. (2004). Issues and challenges in the performance analysis of real disk arrays, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 15, No. 6, 559-574, doi:10.1109/TPDS.2004.9
[26] Simitci, H. (2003). *Storage Network Performance Analysis*, Wiley Publishing, Indianapolis
[27] ALICE Collaboration (2005). *ALICE Technical Design Report of the Computing*, CERN, Geneva
[28] Sundaram, V.; Goyal, P.; Radkov, P.; Shenoy, P. (2003). *Evaluation of object placement techniques in a policy-managed storage system*, Technical report TR#03-38, Department of Computer Science, University of Massachusetts, Amherst
[29] Simitci, H.; Reed, D. A. (1999). Adaptive disk striping for parallel input/output, *Proceedings of the 16th IEEE Symposium on Mass Storage Systems*, 88-102
[30] Chen, P. M.; Patterson, D. A. (1990). Maximizing performance in a striped disk array, *Proceedings of the 17th Annual International Symposium on Computer Architecture*, 322-331
[31] Chen, P. M.; Lee, E. K. (1995). Striping in a RAID level 5 disk array, *Proceedings of the 1995 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, 136-145