DISK DRIVE SIMULATION MODEL DEVELOPMENT

VICKOVIC, L.; CELAR, S. & MUDNIC, E.

Abstract: *This paper presents a detailed development process of discrete event simulation model for a single disk drive. Model development process was conducted in several subsequent phases, and each phase introduced a bit more details, until satisfactory results were achieved. Discussed model is customized for workload imposed by the requirements of the ALICE transient storage system, or more precisely, sequential storing and reading of large data files. With this limitation, simulated throughput differed from the measured one about 0.98 % and 0.8 % for storage and reading, respectively.*

Key words: *discrete-event simulation, disk simulation, disk cache model, queuing delay*

Authors´ data: Dr. **Vickovic**, L[inda]; Prof. **Celar** S[tipe]; Dr. **Mudnic** E[ugen], University of Split, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, R. Boskovica bb, 21000 Split, Croatia, linda.vickovic@cern.hr, stipe.celar@fesb.hr, eugen.mudnic@fesb.hr

This Publication has to be referred as: Vickovic, L[inda]; Celar S[tipe]; Mudnic E[ugen] (2010). Disk Drive Simulation Model Development, Chapter 47 in DAAAM International Scientific Book 2010, pp. 535-548, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-901509-74-2, ISSN 1726-9687, Vienna, Austria
DOI: 10.2507/daaam.scibook.2010.47

1. Introduction

Most modern data intensive applications, like scientific data logging in high-energy physics, use some disk based storage media, like Redundant Array of Independent/Inexpensive Disks (RAID) or Storage Area Network (SAN). As a result it becomes an interesting issue to analyze and improve their performance. In order to perform that, several approaches are possible. One of them, described in (Bokhari et al., 2006), is to perform a set of measurements on the existing storage system. It is advisable, if storage system really exists, but in many cases it can be too expensive in the terms of required hardware and measurements software. The other approach is to develop a model of the system and study it, as a surrogate for system optimization.

Although model development is time-consuming process, once when it is developed it can be used for many purposes, including cost-performance analysis, capacity planning, and similar what-if analysis. Also, optimal system configuration can be chosen simply by changing system components, their organization or input parameters without any additional expenses. Thus, modelling can be used both as an analysis tool for predicting the effect of changes to existing systems and as a design tool to predict the performance of new systems under varying sets of circumstances (Banks et al., 2005).

As described in (Uysal et al., 2001) a model of a complex storage system should be produced by applying a hierarchical decomposition of its relevant parts. As a result, the system is presented as a set of interconnected components, where each represents a part of a system like RAIDs or even disks, switches, routers, etc.

The approach presented in this paper combines a hierarchical decomposition with the "bottom up" approach. This way, from the beginning, the complex storage system is not treated as a whole, but instead the focus is set on the lowest storage component, the disk drive. So, the first step is: a model development for elementary storage component - a single disk drive.

Till now, many disk drive models have been developed and described in the literature. Some of the analytical ones model the behaviour of the whole disk drive. like one described in (Wan et al. 2008), while others are more concerned with some part of the system like I/O response time analyses presented in (Feng et al., 2003).

Nevertheless, according to (Ruemmler & Wilkes, 1994) because of their nonlinear, state-dependent behaviour, disk drives cannot be modelled analytically with any accuracy, so most work in this area uses simulation. Those simulators can be used for various purposes like: generation of disk drive service time distribution described in (Garcia et al., 2008) or for investigation of object-based storage devices (Xi et al., 2006). Also, two functional simulation environments for a disk sub-system have been developed: DiskSim (Ganger & Patt, 1998) and Pantheon (Wilkes, 1995). Both of them include disk modules which have been carefully validated against real disks.

This paper presents a development process for discrete event (DE) simulation model of a single disk drive, created in Ptolemy (Bhattacharyya et al., 1997) an environment for simulation and prototyping of heterogeneous systems developed at Berkeley, the University of California.

A verified and validated model of disk drive is going to be used as an elementary storage module for simulation of complex storage system. The aim is to explore performance of different storage media for the requirements of the ALICE experiment at CERN. The ALICE, one of the four experiments at the Large Hadron Collider (LHC) at CERN is characterized by the most extensive data flow toward data storage. As such, it sets up very high demands on storage system performances, and a model is going to be used for testing and optimization of possible storage architectures.

The remainder of the document is organized as follows. The next section describes workload specification imposed by the ALICE experiment. The system architecture for test measurements, used for model verification and validation is given in section 3. Section 4 contains model development process for disk drive. Developed model verification and validation is discussed in section 5. Finally, conclusions and plans for future development are given in section 6.

2. Workload Specifications

The ability of any model to identify optimal configuration parameters does not just depend on the accuracy of the model, but also on the representativeness of the workload (Simitci, 2003). The workload definition is a bit more complex, and even harder to be properly described than a system itself because it should exactly describe the process of the request arrival to the system components, for given simulation.

The workload for the model presented in this paper was imposed by the requirements of the ALICE Transient Data Storage (TDS) system. To understand those requirements Fig. 1. presents the data flow in the ALICE. It starts from detectors, continues through Data Acquisition System (DAQ) and High Level Trigger and finishes with storage in Mass Storage System (MSS).

The MSS is organized in two levels. The first one is responsible for the real-time data storage and is called the Transient Data Storage (TDS). It consists of arrays, and ensures the required input throughput from DAQ, but has small capacity. The second level, the so-called Permanent Data Storage (PDS) cannot assure required input bandwidth but it provides enough storage capacity for all data acquired during the lifetime of the experiment (ALICE Collaboration, 2005). Such requirements imposed relatively simple workload model for the simulation: a sequential storage or reading of large data files, where all files are of the same size.

3. System Architecture for Test Measurements

One way to validate model behaviour is to use an already verified, available model. The other one is to compare simulation results with measurements done on real, experimental system like in (Bachmat & Schindler, 2002; Uysal et al., 2001). This approach was used for verification and validation of a model described in this paper.

Test measurements were done on a Maxtor MaXLine Plus II disk. Disk was dedicated just for the tests, without any operating system installed. As a result, the whole data transfer from and to the storage media was only produced by the performance measurements program. The program was a standalone client running in the background and it constantly wrote or read fixed length records filled with random data until the predefined output file size was reached. In the same time the exact duration of each transfer was monitored.

The resulting throughput was measured for different file sizes (30, 100, 300, 1000, 2000 MB), and each file size was tested with several record sizes (8, 32, 128, 512, 2048, 8192, 32768, 131072 kB). Also, to get as precise results as possible measurements for different file - record combinations were repeated 15 times. However in the rest of the paper mainly measurement and simulation results for 2 GB files are going to be used, as they are the most likely going to be used in practice. Only in the Section 5 developed model is going to be verified for all file sizes.

4. Disk drive simulation model

Computer systems are usually modelled as discrete event models because their states are generally discrete, like the number of jobs in a queue, available buffer size, and so on. Therefore, the disk drive model simulation elaborated in this paper was modelled as discrete event. Its structure in Ptolemy, described in details in (Vickovic, 2007), is shown in Fig. 2. The model is built from two discrete event components: server and disk module. *Server* module is modelled like a simple delay server. It produces sequential write or read requests toward *Disk* module and presents throughput results when a given task is accomplished. Actually, it simulates the behaviour of the performance measurements program or data storage requests from the ALICE DAQ and data reading requests from PDS.

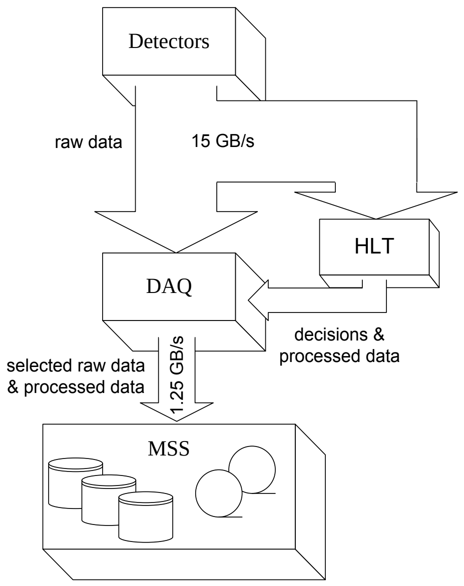
Detectors
raw data
15 GB/s
HLT
DAQ
decisions & processed data
selected raw data & processed data
1.25 GB/s
MSS

Fig. 1. The ALICE data flow

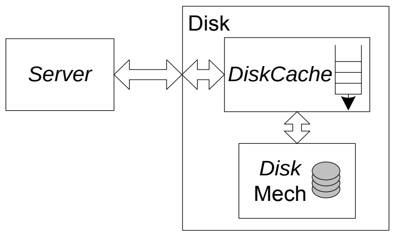

Fig. 2. The general scheme of a disk drive model

Disk module simulates the behaviour of disk drive and it comprises two modules *diskCache* and *diskMech*. The first of them represents the behaviour of disk cache and it was initially modelled like simple First In First Out (FIFO) queue. *diskMech* module models the behaviour of disk drive mechanical components.

Data flow in this model starts from *Server* module which sends request to *Disk*. If it is a write request then it is stored in *diskCache* and the time required for storage is defined by the disk interface speed. When the request is stored, the amount of free space on *diskCache* is decreased and it is pushed on cache queue. From that point *diskCache* leads two independent communications. The former is toward *Server*, which sends another request and their communication proceeds until all requests are sent, or until there is no more free space in *diskCache*. The latter communication is between *diskCache* and *diskMech*. According to FIFO principle, one by one request from cache is sent to *diskMech*. Once when the record is actually stored, the amount of free space in *diskCache* is increased. The communication proceeds until all records are stored.

With these general guidelines, model was developed in several subsequent phases, where each phase introduced a bit more details. Also, model's behaviour was firstly explored for write workload, until satisfactory results are achieved and then for read one. Furthermore, the results from each phase are compared with the measurements performed on a real system.

4.1. Deterministic model for write workload

The initial model of a hard disk drive was very straightforward. Cache was modelled as simple FIFO queue, while the behaviour of its mechanical components was completely based on the equation:

$$average\_service\_time = \frac{f}{2} + \frac{r}{2} + media\_transfer\_time + \frac{l}{i} + h \qquad (1)$$

where: f is full-stroke seek time [s], r is full-disk rotation time [s], l is mean request size [B], t is average size of disk tracks [B], i [B/s] is interface transfer speed, h a controller overhead. media_transfer_time parameter from (1) was not counted analytically but from the linear change of the internal transfer rate. According to the data sheets given by manufacturers, the disk internal transfer rate is maximal at the outer most track and minimal at the inner most track. So, assumed that disk is empty, the storage can start from the outer most track with maximal speed and then decreased by some Δspeed value, each time the track on disk is changed. Minimal value is reached at the inner most track when the disk is full.

The disk drive was presented like a two dimensional array where the number of rows is equal to the number of cylinders, and the number of columns to the number of disk surfaces. In this case whenever it is necessary to switch the disk head (change a column) or switch to another cylinder (change a raw) an additional delay is counted. Those delays, produced by switching the disk head or changing to the subsequent disk cylinder, are given in disk's technical specifications.

Not only that average difference between the measured and simulated results for this model was around 17 %, but it also showed descending trend for large record sizes (32768, 131072 kB) that was not a case for measurements. An idea to solve this problem was found in (Shriver 1997) where some additional cache service time delay was added in the model.

To explore this idea, two different delays were tested. 0.001ms delay lowered the simulated system throughput slightly, but the general descending trend for large files was still present. 0.01ms cache service time delay generated lower throughput for small and medium record sizes, and the throughput curve behaved more like the one for the real measurements.

Also, the test measurements were repeated 15 times to get as precise results as possible, so the next step in model development was to test its behaviour on the amount of data to be written. Like it was expected, the resulting output was affected by the number of files because the internal transfer rate decreases with the amount of stored data. The resulting output is shown in Fig. 3.

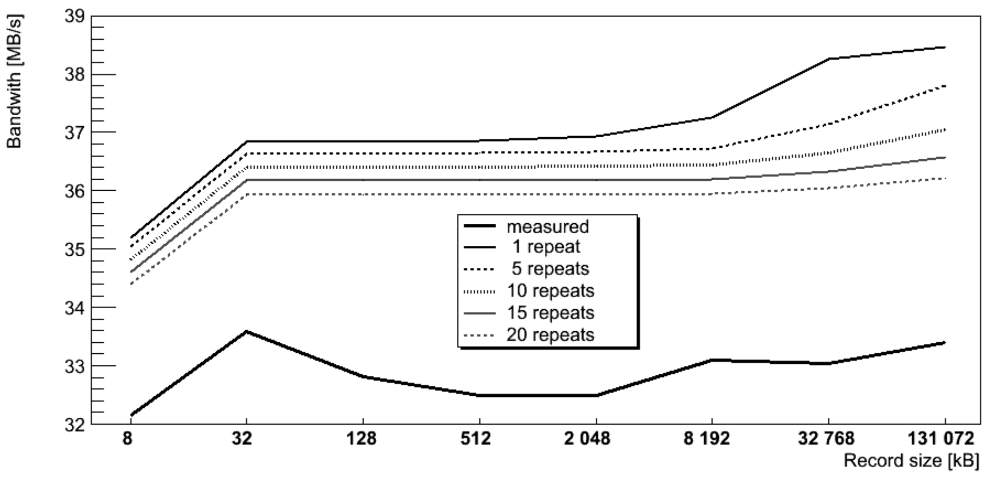

Fig. 3. Simulation output for a linear disk drive model for several repeats

It is easy to conclude that the average throughput gets lower as the number of repeats grows. Although in Fig. 3 the results for 1, 5, 10 and 20 repeats are presented, all single disk measurements were done with 15 repeats. Further on just the simulation output for 1 and 15 repeats will be simulated and compared with measured results.

4.2. Stochastic model for write workload

As the described model of disk drive had not produced satisfactory results, a next step was to introduce some “random” behaviour in the simulation. The most common parameters with random behaviour in disk drive modelling are: workload generation, modelling of disk seek and rotation time (Ruemmler & Wilkes, 1994; Shriver, 1997; Shriver et al., 2000) and service time delay simulation. Although, stochastic elements cannot be used for workload generation, because all real measurements were done with just one sequential data stream, they can be used for service time delay in two components *Server* and *diskCache*. For service time delay generation three general distributions: normal, uniform and exponential were individually incorporated in the model and tested.

Random service time delay implied in *Server* module referred delay between successive requests sent to the disk drive, but it proved to be immune on service time delay change. Obviously, the delay produced by storing a file is much bigger than a delay between successive requests and it does not influence the system behaviour.

As shown in previous section, *diskCache* module is sensitive to the changes of service time delay and because of that some random service time delay from the interval of standard DRAM cache delay was presented to its outputs. The resulting model's behaviour is presented in Fig. 4. and it can be easily concluded that uniform distribution should be avoided, while the normal and exponential distribution showed similar results. In average the difference between measured and simulated results for normal and exponential distribution was 2.22 % (0.98 % without 8 kB records) and 2.01 %(1.03 % without 8 kB records), respectively. As the largest record sizes are most likely to be used in practice, discrepancy for 8 kB records can be neglected and the normal distribution is chosen.

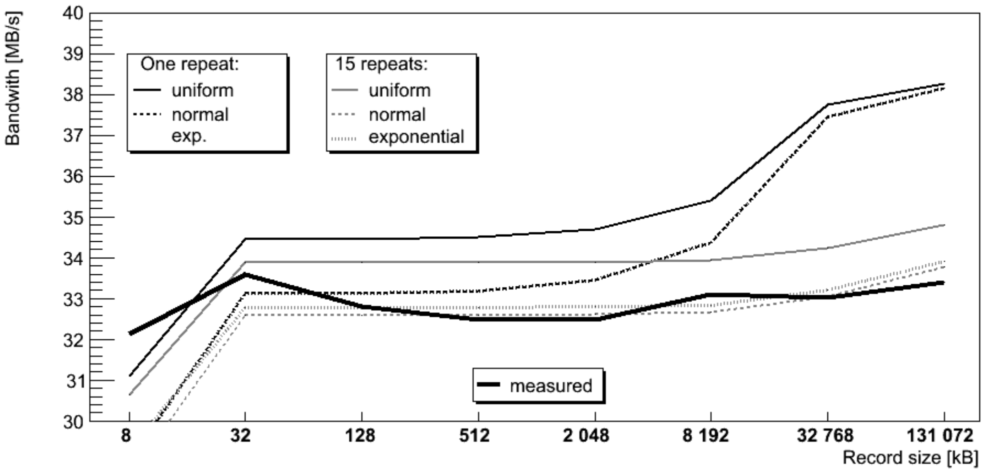

Fig. 4. Simulation output for a different random service time delays in diskCache module

The last component in the model tested with random behaviour is a *diskMech* module. The only way to introduce randomness in this component is to randomly generate the track and the associated speed to start the storage from. All other movements, because of specific workload requirements, include just head and track switching. The influence of randomly generated starting track position is presented in Fig. 5.

Although the difference in throughput increases significantly with distance from the outer most track, it can be neglected if the starting track is near the outer most one. For example, if writing starts from track laying for 2 % away from outer most track the throughput is influenced by 0.7 %. As a result, initial assumption that writing starts from outer most track is going to be used.

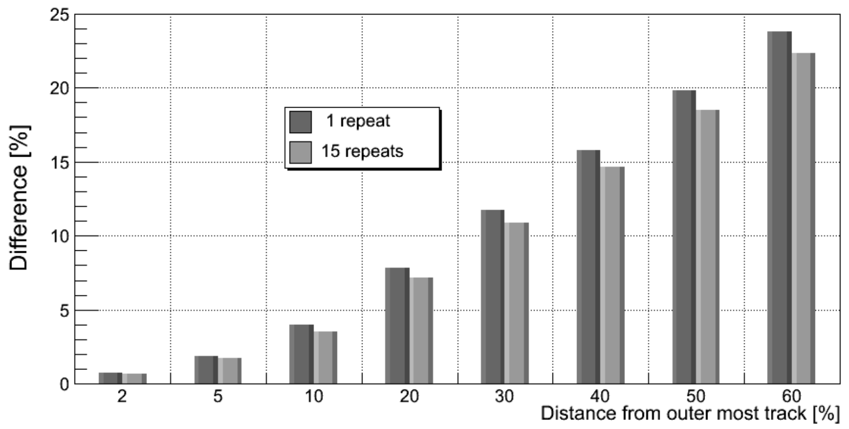

Fig. 5. The influence of the starting track position on the simulation throughput for write workload

The model presented in this section is the most complex one, but it also produced the best results and it is going to be used for further storage system model development.

4.3. Stochastic model for read workload

With the functional model for write workload, the next step in the simulation development was to accommodate that model for the read workload. Although most of the functionalities from the write workload model can be used, different direction of dataflow required some changes.

Once when functional version was achieved it was firstly tested with the stochastic parameters that produced best results for write workload. Simulation output, shown in Fig. 6, has the same increasing trend like one produced by write workload, but difference between measured and simulated results (18.02 % difference for 8 kB record size and 7.7 % difference in average for the other record sizes) pointed out that it was not possible to describe read and write cache behaviour with the same stochastic parameters.

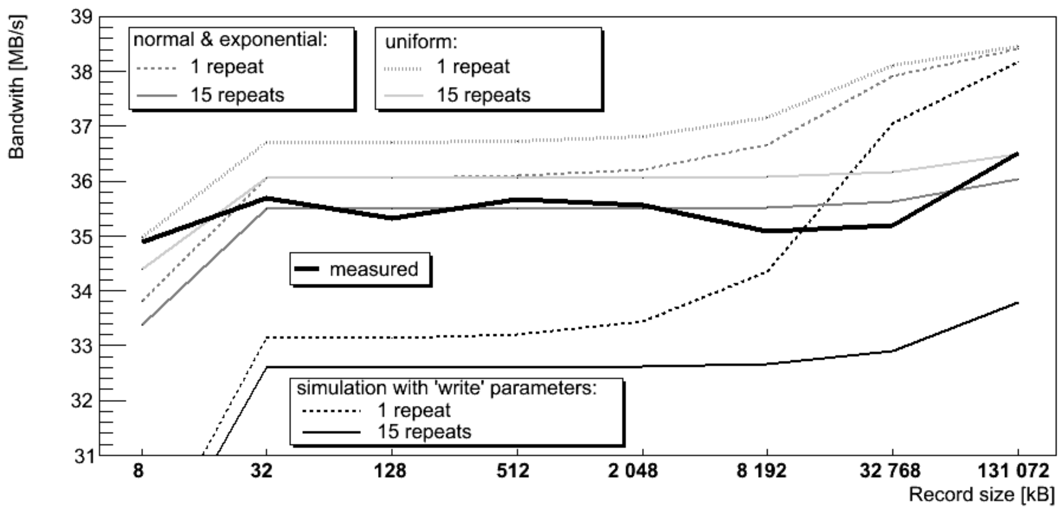

Fig. 6. Simulation output for a read workload

Such outcome was expected because of two reasons, one related with the disk drive mechanism, and other related with the disk cache. For writing, the disk drive mechanism should position read/write head exactly to avoid damage of neighbouring bytes, while for reading such precision is not necessary and because of that reading from disk is a bit faster than writing. Besides, the cache model applied in this simulation is very simple and it does not take into account any throughput increase produced by read ahead cache policy. With the intention to keep a model as simple as possible those two factors are presented by some smaller delay generated by diskCache module. The resulting model’s performances for normal, exponential and uniform distribution are also presented in Fig. 6.

Differences between simulation and measured results for this scenario are shown in Fig. 7. The results are actually good as the majority of differences are in range [-2 %, 2 %]. A bigger difference is only produced for 8 kB record size, where the influence of read ahead, or some internal cache on server side is the most visible. However, it was not a surprise as it was the case for the write workload and the same model was used for both workloads.

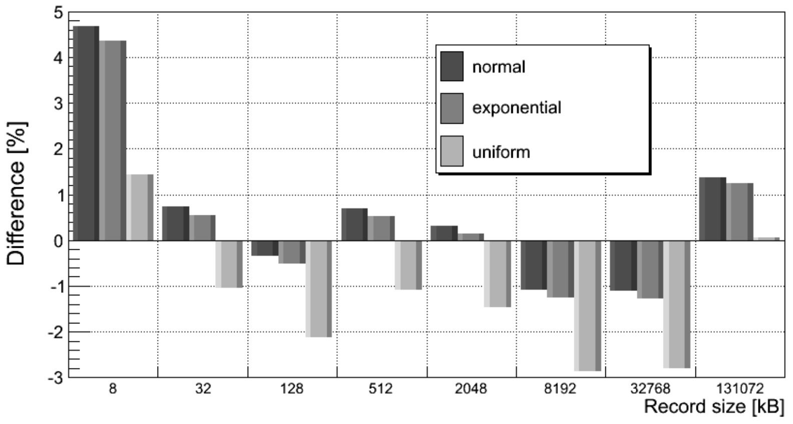

Fig. 7. Differences between measured and simulated results for a random service time in diskCache

Although the greater part of distributions and parameters produced satisfactory results, to describe the service delay produced by the disk cache normal distribution is chosen because it produced the 1.29 % average difference for all record sizes and 0.8 % if 8 KB records are omitted.

5. Model Validation and Verification

The main assumption in the model development was that writing on the disk started from the outer most to inner most track and internal transfer rate is decreased in the process. To verify it, a measurement was preformed where writing had started from empty disk and lasted until it was full. Resulting bandwidth for 1 and 2 GB files are presented in Fig. 8. From the measured results it is obvious that writing to the disk is not a constant process. The results verified assumption under question that writing is really faster in the beginning and getting lower with the amount of data already on disk.

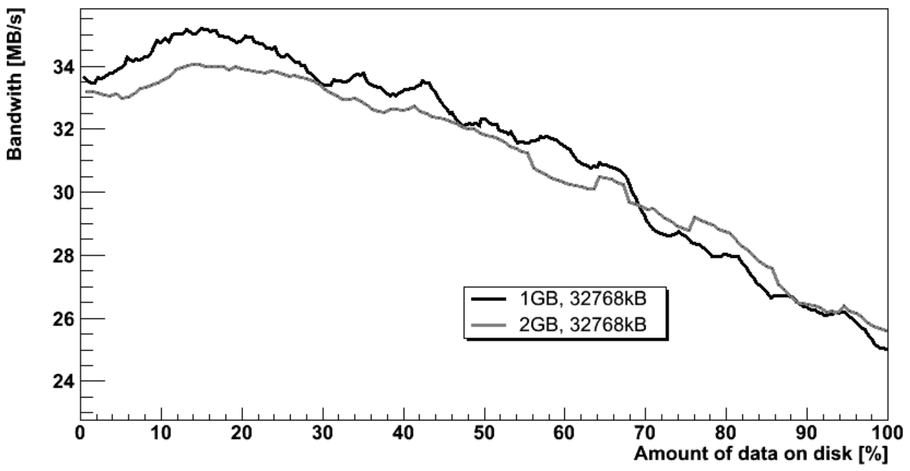

Fig. 8. Measured writing performance of a disk drive

The chosen model for the write workload was calibrated for 2 GB files, so the final step in the model development was to simulate the storage throughput for all other file sizes measured on the real system. Resulting throughputs are presented in Fig. 9.

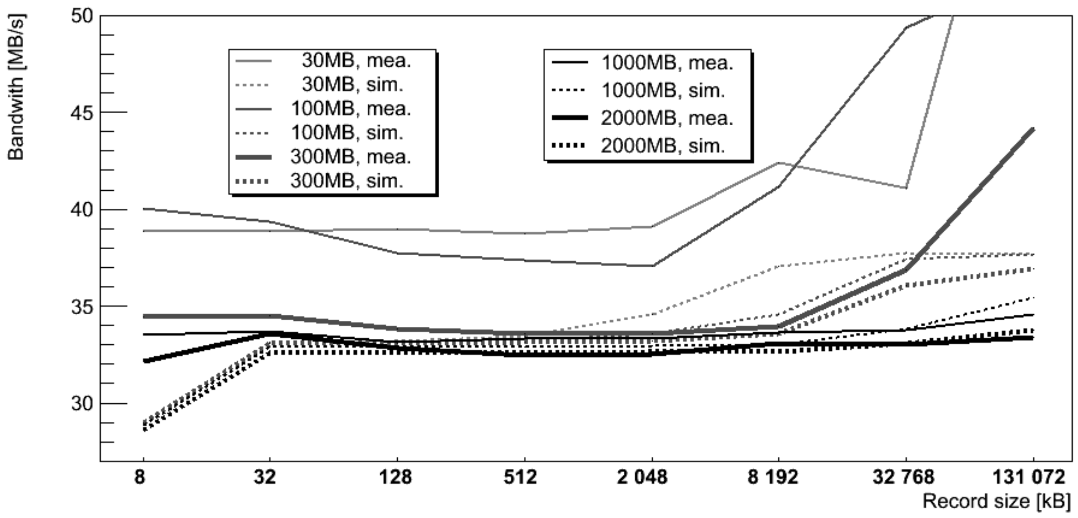

Fig. 9. Simulation and measurements results for write workload

Differences between measured and simulated values are shown in Fig. 10, where values marked with "x" are not measured because those records are bigger than the files themselves. Again, if single record sizes are observed, the biggest difference is produced by 8 kB records, and if it is omitted, the average differences between measured and simulated results for 2 GB, 1 GB and 300 MB are reduced on 0.86 %, 1.52 %, and 4.05 %, respectively.

The model development for read workload was not so extensive, because it is just an adaptation of the write workload model. In practice, reading from disk is a bit faster than writing and this real-life situation in the model is presented with some smaller service time delay produced by the disk cache. The differences between measured and simulated results for all file sizes are shown in Fig. 11.

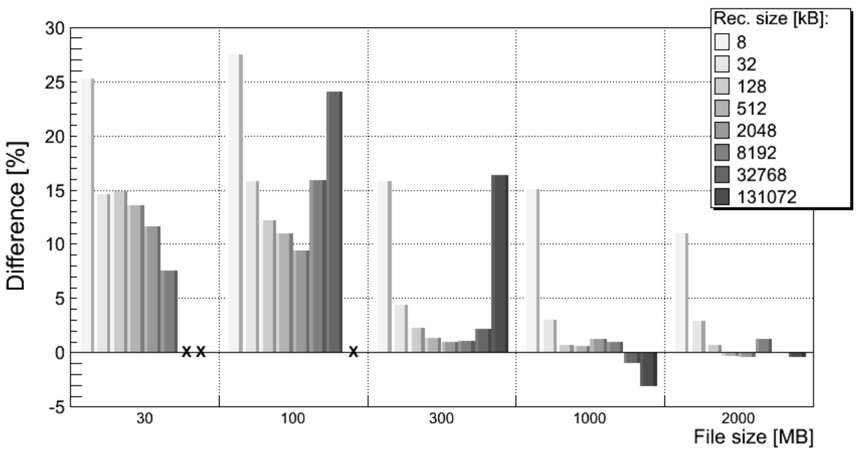

Fig. 10. Differences between measured and simulated results for write workload

At first glance, the results for all file sizes are not encouraging. The average difference for 30 and 100 MB files is around 66.3 %, whereas for 300 MB files it is lower and around 23.4 %.

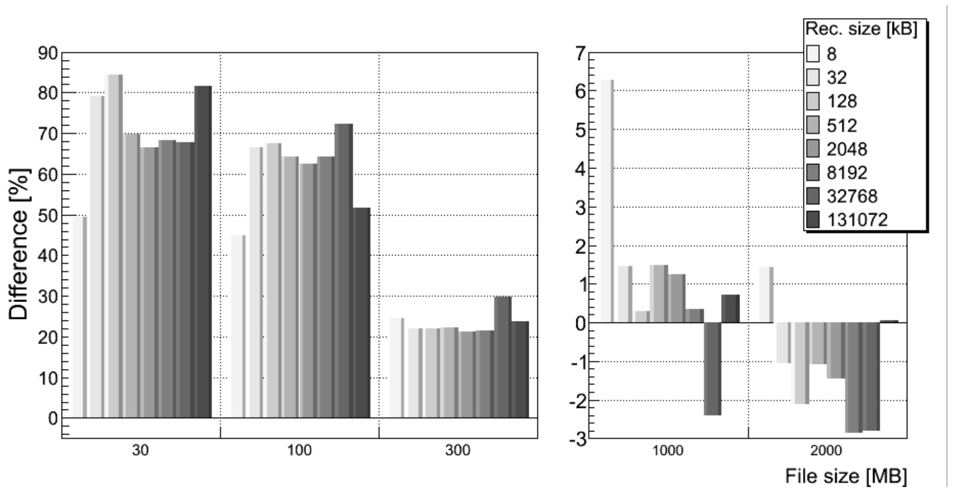

Fig. 11. Differences between measured and simulated results for all file and record sizes with read workload

Only for 1 and 2 GB files it is acceptable and around 1.7 %. In this case it is interesting to see the results of the real measurements and they are shown in Fig. 12. Special attention should be paid on throughput for 30 and 100 MB. It is even faster than the ATA interface (133 MB/s), used to connect the disk drive to the server. In conclusion the measurement results for 30 and 100 MB file sizes are not completely reliable for the simulation, because data were apparently stored in some internal cache on the server side. For 300 MB files the influence of the internal caching is still visible because the throughput is still larger than the specified disk internal transfer rate (24-42 MB/s), but obviously internal cache was not big enough to store all data.

The developed model clearly does not provide satisfactory results for small and medium file sizes with read workload. But its main purpose is the simulation of the ALICE transient data storage system, which deals mostly with large files (events are grouped in 2 GB files), so the small and medium file sizes are negligible for this, specific kind of workloads. Additional changes aiming to complete mimicking the real cache behaviour and file system influence would introduce many changes in the model and necessarily make it much more complicated.

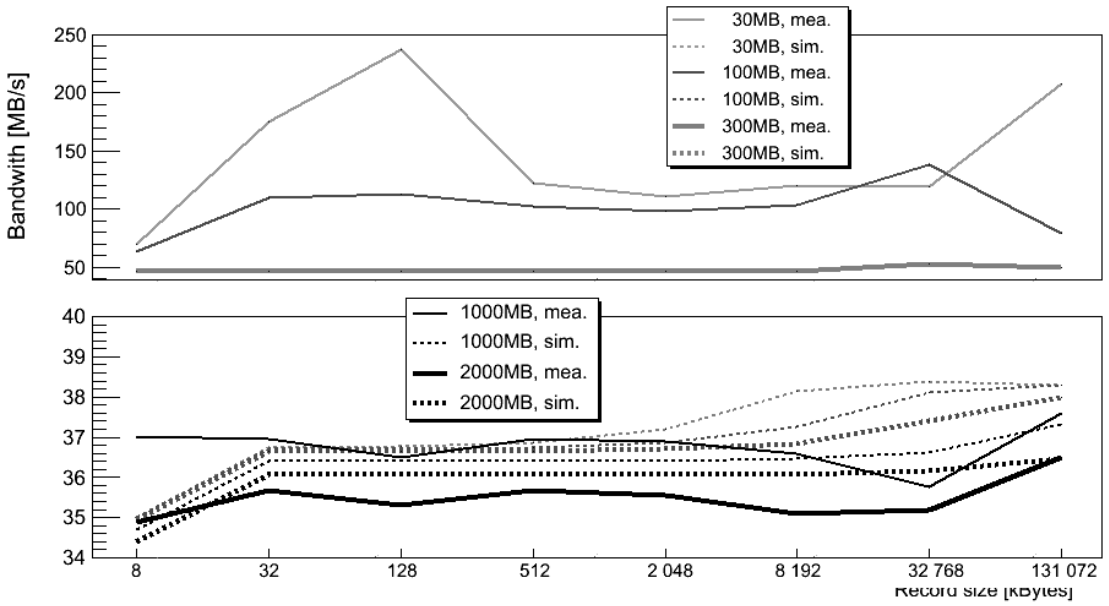

Fig. 12. Disk simulation and measurements results for different file sizes with read workload

6. Conclusions and Future Work

This paper gives an extensive and detailed preview of disk drive model development. Although such models have existed for more than 15 years, they are still under study as a part of more complex storage systems like RAIDs and SANs.

The novelty presented by this model is its simplicity achieved by introduction of analytically counted positioning time for disk drive mechanism (seek time and rotational latency) in the simulation. As a result the model can be treated as hybrid one, while all other models presented in literature were strictly analytical or simulation ones. This paper also gives a detailed study how much additional service time delay in individual modules can influence the behaviour of the whole simulation.

Although in the beginning the modelling of the disk mechanism seemed much more complex than the modelling of the other system components, especially the disk cache, it proved to be a wrong assumption. Regardless of disk internal rate was counted, it did not influence simulation as much as introduction of the cache service time delay.

It is also important to notice that discussed model of disk drive is customized for the requirements of the ALICE transient storage system, which means it is appropriate for sequential storing and reading of large data files. So, if this model is used just for simulations with this specific workload requirements, it provides more than satisfactory results, about 0.98 % for write and 0.8 % for the read workload without 8 kB records. Even if they are included in average difference, the results are still acceptable: 2.22 % for write and 1.29 % for the read workload. The lowest difference between measured and simulated results for available models published in literature (Ruemmler & Wilkes, 1994; Thekkath et al., 1994) till now is around 5 %.

Future development of presented model can be extended in two directions. The first one is to adapt existing model to produce better results for small record sizes by modelling the influence of file system and caching in *Server* module.

The other one is to use existing model as an elementary module for RAID simulation and then to develop a model of SAN with the goal to explore the influence of different parameters on the ALICE transient data storage system.

7. References

ALICE Collaboration (2005), *ALICE Technical Design Report of the Computing*, CERN Desktop publishing, ISBN 92-9083-247-9, Geneva, Switzerland

Bachmat, E. & Schindler, J. (2002). Analysis of methods for scheduling low priority disk drive tasks, *Proceedings of the 2002 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, Muntz, R. (Ed.), pp. 55-65, ISBN 1-58113-531-9, Marina del Rey, CA, USA, June 2002, ACM New York, NY, USA

Banks, J.; Carson, J. S. II; Nelson, B. L. & Nicol, D. M. (2005). *Discrete-Event System Simulation*, Prentice Hall International, ISBN 9780131446793, New York, USA

Bhattacharyya, S. et al (1997). The Almagest – Ptolemy 0.7 User's Manual, University of California at Berkeley, CA, USA

Bokhari, S.; Rutt, B. & Wyckoff, P. (2006). Experimental analysis of a mass storage system, *Concurrency and Computation: Practice and Experience*, 18,15, (December 2006) pp. 1929–1950, ISSN 1532-0634

Buck, J.; Ha, S.; Lee, E. & Messerschmitt, D. (1994). Ptolemy: A Framework for Simulating and Prototyping Heterogeneous Systems, *International Journal of Computer Simulation, special issue on Simulation Software Development*, 4, 2, (August 2004) pp. 155-182, ISSN 0360-0300

Feng, D.; Jiang, H. & Zhu, Y. (2003). I/O Response time in a fault tolerant parallel virtual File system, Technical Report, Department of Computer Science and Engineering, University of Nebraska-Lincoln, USA

Ganger, G. R. & Patt, Y. N. (1998). Using system-level models to evaluate I/O subsystem designs, *IEEE Transactions on Computers*, 47, 6, (June 1998) pp. 667–678, ISSN 0018-9340

Garcia, J.D.; Prada, L.; Fernandez, J.; Nunez, A. & Carretero, J. (2008). Using Black-Box Modeling Techniques for Modern Disk Drives Service Time Simulation, *Proceedings of 41st Annual Simulation Symposium (ANSS 2008)*, pp. 139 – 145, ISBN 0-7695-3143-1, Ottawa, Canada, April 2008, IEEE Computer Society

Ruemmler, C. & Wilkes, J. (1994). An introduction to disk drive modelling, *IEEE Computer*, 27, 3, (March 1994) pp. 17-28, ISSN 0018-9162

Shriver E. (1997). Performance modelling for realistic storage devices, PhD Thesis, New York University, New York, USA

Shriver, E.; Hillyer, B. & Silberschatz, A. (2000). Performance Analysis of Storage system. Published as a section in Performance Evaluation: Origins and Directions, Lecture Notes in Computer Science in New York University Computer Science, pp. 33-50, Springer Verlag

Simitci, H. (2003), *Storage Network Performance Analysis.* Wiley Publishing, Inc, ISBN 978-0-7645-1685-6, Indianapolis, USA

Thekkath C.A.; Wilkes J. & Lazowska, E.D. (1994). Techniques for File System Simulation, *Software Practice and Experience*, 24, 11, (November 1994) pp. 981–999, ISSN: 1097-024X

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 (MASCOTS-2001),* pp. 183-192, ISBN 0-7695-1315-8, Washington, DC, USA, August 2001, IEEE Computer Society

Vicković, L. (2007). Management and optimization of mass data storage system for ALICE experiment, PhD thesis, University of Split, Split, Croatia

Wan, F.; Dingle, N.J.; Knottenbelt, W.J. & Lebrecht, A.S. (2008). Simulation and modelling of raid 0 system performance, *Proceedings of the 22nd Annual European Simulation and Modelling Conference (ESM'08*), Bertelle, C. & Ayesh, A. (Ed.), pp. 145-149, ISBN 978-90-77381-44-1, October 2007, Le Havre, France, Eurosis, Ostend, Belgium

Wilkes, J. 1995. The Pantheon storage-system simulator, Technical Report HPL-SSP-95-14, HP Laboratories, Palo Alto, CA, USA

Xi, W.; For W.; Wang D.; Kanagavelu R. & Goh W. (2006). OSDsim - a Simulation and Design Platform of an Object-based Storage Device, *Proceedings of the 23rd IEEE Conference on Mass Storage Systems and Technologies (MSST2006),* Kobler, B. &Hariharan, P.C. (Ed,), pp. 97-102, College Park, Maryland, USA, IEEE Computer Society