Computing and Informatics, Vol. 31, 2012, 1345–1366

SEMANTIC-BASED STORAGE QOS MANAGEMENT METHODOLOGY – CASE STUDY FOR DISTRIBUTED ENVIRONMENTS

Darin Nikolow

AGH University of Science and Technology
Faculty of Electrical Engineering, Automatics, Computer Science and Electronics
Department of Computer Science
al. A. Mickiewicza 30, 30-059 Krakow, Poland
e-mail: darin@agh.edu.pl

Communicated by Jacek Kitowski

Abstract. The distributed computing environments, e.g. clouds, often deal with huge amounts of data, which constantly increase. The global growth of data is caused by ubiquitous personal devices, enterprise and scientific applications, etc. As the size of data grows new challenges are emerging in the context of storage management. Modern data and storage resource management systems need to face wide range of problems – minimizing energy consumption (green data centers), optimizing resource usage, throughput and capacity, data availability, security and legal issues, scalability. In addition users or their applications can have QoS (Quality of Service) requirements concerning the storage access, which further complicates the management. To cope with this problem a common mass storage system model taking into account the performance aspects of a storage system becomes a necessity. The model described with semantic technologies brings a semantic interoperability between the system components. In this paper we describe our approach at data management with QoS based on the developed models as a case study for distributed environments.

Keywords: Data management, service level agreement, quality of service, data storage, storage system

1 INTRODUCTION

According to the IDC report [1] the size of the digital universe[1] in 2009 was about 800 EB and was expected to grow to 1.2 ZB till the end of 2010. 80 % of the digital universe is unstructured data[2]. The growth of data is caused by ubiquitous personal applications and devices like mobile phones, digital cameras, etc, and also by enterprise and scientific applications. The last ones can generate huge quantities of data. Examples of data intensive scientific applications come from the field of physics, e.g., HEP experiments [2], astronomy – e.g. LWA (Long Wave Array) telescope [3], bioinformatics [4], Earth science [5], etc. There are also classical storage consuming applications like backup and archiving [6].

As the size of data grows, the problem of efficient data management arises. Methods of data management can be quite different depending on the nature of data. The nature of data can be described by attributes such as: size, persistence, durability, reproductivity (the ability to be regenerated), access pattern (WORM[3], R/W[4], WORR[5]), confidentiality, importance, etc. For example some data need to be kept for a long time or even forever (backup and archiving), while other need to be accessible (online access) – one method for data management is used for backup and archiving, while another is used for online data.

Modern data and storage resource management systems need to face wide range of problems – minimizing energy consumption (green data centers [7]) on one hand, and optimizing resource usage, throughput and capacity on the other hand [8]. Other problems which need to be considered concern data availability, scalability, security and legal issues [10]. In addition, users or their applications can have QoS (Quality of Service) requirements concerning the storage access, which further complicates the management [9]. Typical QoS requirements for data storage, called further storage QoS requirements, are: data access latency, data access transfer rate, storage space.

The storage QoS management term used further in the article means storage resource management for achieving QoS for data access. The managed storage resources are usually complete storage systems like HSM (Hierarchical Storage Management) systems or disk arrays but can also refer to a particular part of the storage system, like a disk partition.

In order to manage the storage QoS and especially the storage performance, some information about the current state of the managed storage system is necessary. There are plenty of various storage systems used in the IT world. These storage systems come from various vendors, have different architectures and user/diagnostic interfaces. That is why obtaining the data (as a list of possibly dynamically changing

[1] digital universe represents all the data created, stored and replicated in the world

[2] data that either does not have a pre-defined data model and/or does not fit well into relational tables

[3] Write Once Read Many

[4] Read/Write

[5] Write Once Read Rarely

parameters) describing the current state of a storage system is system dependent. On the other hand, the values of storage QoS metrics are specified by the user and for that reason the QoS metrics should be defined in higher abstraction level independent on storage system used by the service provider.

To cope with this problem a common mass storage system model taking into account the performance aspects of a storage system need to be developed. The model could be used to describe the state of a storage system, which can be taken into account when making storage resource management decisions. The model could be described using semantic technologies, which is expected to bring a semantic interoperability between the other components of the system and the services themselves. Modeling of computer system has been studied by many researchers resulting in specifying models like CIM (Common Information Model) [11], GLUE (Grid Laboratory Uniform Environment) [12], Rome [13], but the current trends concerning the dynamically changing distributed environments, the virtualization of computing resources and the complexity of management for QoS bring new challenges for the modeling.

In the presented approach to this problem we propose methodology for building storage management system with QoS in mind using the following components:

- common mass storage system model,
- ontologies based on the model,
- set of sensors for monitoring the storage systems performance and delivering the relevant parameters defined in the model,
- set of estimators for estimating storage systems performance, which can be used for building higher data management layers respecting storage QoS.

In this paper we describe our approach to data management with QoS and we present use cases, in which the proposed methodology has been used. These use cases concern:

- intelligent monitoring of storage performance used for administration purposes,
- best replica selection and best storage location selection in distributed storage environment, in which replication techniques are used to increase data protection and availability,
- SLA (Service Level Agreement) metrics monitoring for storage services,
- acceleration of distributed computations for data intensive applications in distributed environments.

The rest of the paper is organized as follows. The next section presents the state of the art. The third section presents the proposed methodology. The forth section discusses the use cases implemented by using the methodology and the last section concludes the paper.

2 STATE OF THE ART

There are many research papers in the field of storage QoS management. These studies cover wide range of related topics, like: computer system modeling; resource management (including storage) in distributed computing environments; managing and monitoring of SLA; automated replica management in distributed environments for efficient data access; performance monitoring and prediction. Selected studies on the mentioned subjects are presented below.

There are models, like CIM (Common Information Model) or GLUE (Grid Laboratory Uniform Environment), which can be used for representing computer systems. The CIM [11] is an open standard provided by the DMTF (Distributed Management Task Force). It is an object oriented model of management elements used in the IT environment. The GLUE [12] is an information model of Grid environments for providing interoperability between grid components. There are lacks in these models concerning the representation of performance related details, which are necessary for predicting the performance of storage systems.

In [13] an information model for describing storage systems QoS requirements is presented. The model is object-based and is used to glue together storage system designs, monitoring and configuration. The paper describes a complete automated storage management system and specifies its QoS parameters. In our case we do not built a complete system but rather focus on extending the storage resource model and making use of semantic technologies to allow precise modeling of storage resources and flexibility in building storage management systems.

Grids and clouds are distributed computing paradigms following the idea of providing of computing and storage resources as an utility to the end user. Many problems need to be solved in order to provide the computing power as utility [14]. Some of these problems are related to storage and management of data. When using computing power as utility, certain parameters concerning its quality need to be guaranteed, e.g., the amount of Gflops/hour (similar to kW/h for electric energy), storage bandwidth, etc. Providing storage QoS for such distributed multi-user environments with shared resources is a challenging task. The user requirements concerning the QoS are specified in the SLA [14]. The efficient resource management respecting the SLA is a challenging task especially in the context of storage systems.

In [15] an approach to execution management of grid application and enforcement of the SLA contractual terms is presented. The monitoring service used in this approach monitors only basic parameters, while the storage performance related parameters are not taken into account.

In [16] the DesicionQoS system is proposed. It aims at arbitrating storage resources among clients and enforcing QoS. It relies on machine learning techniques to retrieve information about the storage system workloads and their correlations. Every single I/O operation is monitored.

In [17] an approach to automated performance control in a virtual storage system called Storage@desk is presented. The system uses idle storage resources on desktop

machines for constructing iSCSI storage volumes, which are then provided to the clients. Replication is used to achieve the desired level of data availability.

In [18] a solution for QoS differentiation for disk array systems is proposed. It allows for QoS management where certain I/O streams are throttled in order to favor more important I/O streams.

In [19] a system for real-time data capture with QoS guarantees is presented. The system is designed for the WORR type of data and uses a ring buffer approach keeping the data on disk until they are analyzed. The study is promising aiming at creating a filesystem with QoS guarantees.

In order to provide efficient storage service with QoS a proper monitoring and data access time estimation is necessary, as can be seen from the above examples. Adopting semantic techniques to the monitoring and estimation layer could improve the organization of the higher storage management layers. These reasons have motivated us to start this research.

3 STORAGE MANAGEMENT METHODOLOGY

The storage management methodology proposed in this paper provides models (including semantic models) of storage systems and their implementation in the form of sensors and estimators allowing for building data and storage management systems with QoS in mind. The methodology specifies the building blocks (see Figure 1) of such systems – common storage system model – C2SM, ontologies based on the model, sensors for monitoring and estimators for prediction of storage system performance. These building blocks are presented in the following sections.

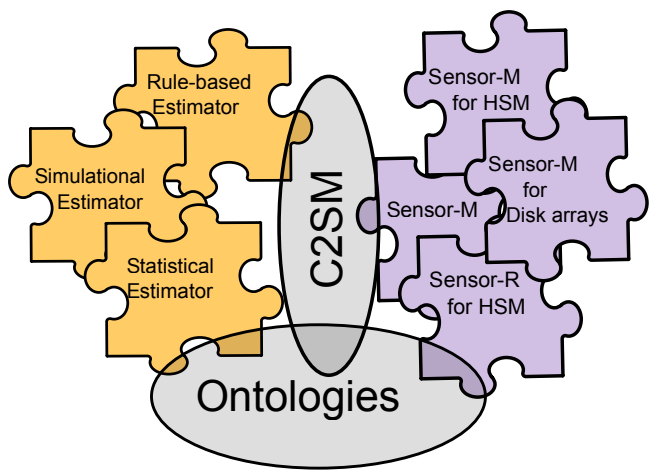


Fig. 1. Semantic-based storage QoS management methodology's building blocks

3.1 C2SM – A CIM Based Common Mass Storage System Model

There are many storage systems used today, like HSM systems, disk arrays, disk based file systems, cluster filesystems. These systems differ from each other in terms of architectural design, functionalities and interfaces.

HSM systems are complex systems, which may contain other storage systems and devices. A typical HSM system consists of one or more tape library, disk cache (which can reside on a disk array) and a server running the HSM software. There are various HSM systems, e.g., EMC DiskXtender, HP FSE, CERN CASTOR, IBM TSM. These systems have different sets of diagnostic/administration/management utilities or APIs but also hold similar parts and functionalities. The same rule applies for disk arrays, which generally consists of a bunch of disks, cache memory and interface controller.

The C2SM has been proposed to cope with the heterogeneity of storage systems bringing a uniform way of representing such systems.

The proposed common mass storage system model consists of two parts: storage system state model and state transition algorithms. The state model is defined as a set of well defined storage system related parameters, which are able to describe, with a certain level of detail, the state of any storage system including HSM systems, disk arrays and cluster file systems. The parameters concern such aspects of storage systems like performance, capacity, location of data and media, system behavior. The state transition algorithms describe how and when the state of a modeled storage system is changed.

The model plays an important role in our methodology providing an unified way of describing the state of storage system used by the other elements.

3.1.1 Storage System State Model

The model is based on the standard CIM (Common Information Model) [11]. Relevant classes, not available in the current version of CIM, have been added for describing these elements and parameters, which are essential for our model. The classes are shown in Figure 2.

We have added the *AGH_Storage_System* class, which is our main class containing parameters common for any storage system like storage capacity and transfer rate. Most of the classes in our model describe HSM system elements, represented by the *AGH_HSMSystem*, since these systems are more complicated than the other kinds of storage system described in the model – disk based storage systems. Here we can find such elements like media libraries, media changers, drives, disk partitions.

3.1.2 State Transition Algorithms

Two types of state transition algorithms describing the behavior of the modeled storage systems are assumed:

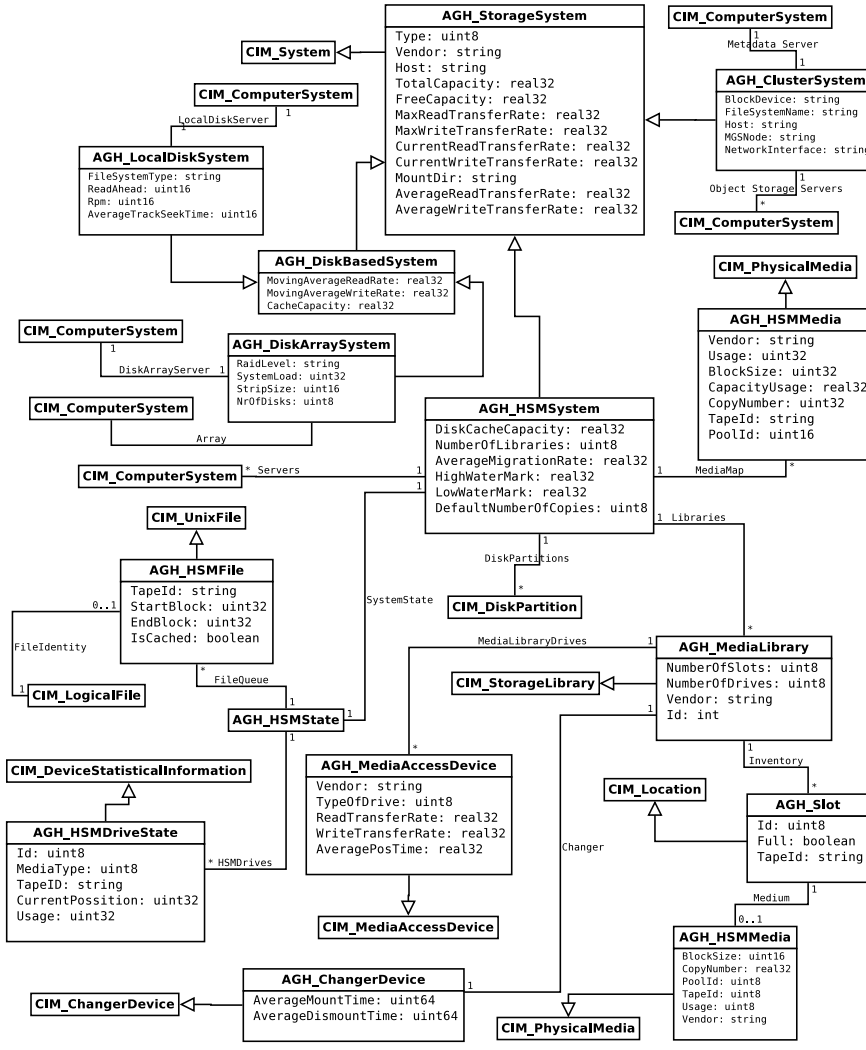


Fig. 2. C2SM class diagram

- black box algorithms used for disk based systems [20],
- grey box algorithms used for HSM systems [21].

These algorithms are general for the given kind of storage system being modeled (HSM, disk array, local disk) and need to be tuned for a particular storage system. The tuning relies on specifying its essential performance related parameters independent on the current storage system state. These parameters can be gathered from the vendor's product specification or obtained experimentally by running

benchmarks. Below we describe the state transition algorithms for HSM systems in which the grey box approach is used, and algorithms for disk arrays where the black box approach is used.

The access time for HSM systems can vary a lot depending on the location of data – from milliseconds, when the data is cached, to tens of minutes when the data is vaulted. For this kind of systems it seems feasible to implement more precise, but slower, estimators based on simulation. In this case the grey box approach seems to be the best solution.

For the HSM systems, we assume that the general HSM system consists of disk cache, at least one automated media library, or stand alone removable media drive and at least one computer running the HSM software. When writing data, the whole file is transferred to the disk cache and later it is copied to a removable medium, like a tape or an optical disc. In order to copy the file to the removable medium the medium has to be mounted in a drive and then prepared for write transfer, by positioning the head in the case of tape media.

When reading data two cases are possible – the file is in the disk cache or the file resides on a removable medium only. In the first case the data is just transferred from the cache, while in the second case the file must be copied back from the removable medium to the disk cache and only then transferred. This process usually involves: waiting for a resource (optical disk, tape or drive) dismounting and mounting a medium, positioning, transfer. Positioning of a tape drive is also modeled since it can take considerably long time. More details about the state transition diagram used can be found in [21].

The disk arrays have much lower latencies than the HSM systems and therefore the modeling should be simpler for faster obtaining of results. Therefore, the black box approach used here is appropriate. The black box approach assumes that we do not know anything about the internal architecture of the storage system being modeled. For the disk arrays the algorithm takes into account the existence of cache memory and its influence on the data transfer. This influence is manifested in such a way that for files smaller than the cache memory the transfer is fast, while for bigger files the transfer is slower [20]. The model of the disk array needs to be tuned by measuring essential parameters being part of it (the model). The model describes these parameters and how to use them to calculate the performance parameters. These parameters are: the maximal transfer rate for the host connection, the maximal transfer rate for the RAID disk pool, the minimal size of file to achieve the maximal transfer rate and the maximal file size after which the performance gets degraded.

3.2 Sensors and Estimators

Another building block of our methodology are sensors and estimators. The sensors use the state model described in the previous section, while estimators – the state transition algorithms.

3.2.1 Sensors

Sensors provide information about specific performance related parameters of the given storage system being part of the distributed computing environment. Sensors are running on hosts, which provide access to the given storage systems. Sensors are system dependent, since there are different ways of obtaining the relevant parameters depending on the specific storage system. Two types of sensors are introduced – Sensor-M and Sensor-R. The Sensor-M cyclically obtains the required parameters and sends them to a monitoring service, which collects data from many different sensors. In addition to the functionality of Sensor-M, Sensor-R provides on demand some specific parameters, which are not kept in the monitoring service database because of their size and nature. These specific parameters concern the location of a given file, for example the *isCached* attribute of class *AGH_HSMFile* (see Figure 2). These parameters are obtained only when requested. The Sensors-R were implemented by us as RESTful services [22], while the Sensors-M as lite daemons. The sensors send data using common data format, according to the C2SM model. The sensors currently support local disk storage systems, disk array storage systems and the following HSM systems: HP FSE (File System Extender) and IBM TSM (Tivoli Storage Manager). These sensors can be found on [23].

3.2.2 Estimators

Estimators predict future storage performance based on the data obtained from sensors. We have defined three types of estimators depending on the method used for prediction: statistical, rule-based and simulational. Statistical estimator bases on the performance statistics of a given storage system for a certain period of time delivered by the monitoring service. Rule-based estimator predicts the storage performance using predefined rules for checking various conditions and applying one of the predefined formulas for calculating the expected performance. Simulational estimator, which is the most sophisticated one, predicts the performance by simulating the future state of storage system based on the current one. This type of estimators uses the state transition algorithm described in the model (see Section 3.1.2). The estimated performance for a given data access request (for example reading a file) is provided by two values: the data access latency time and data access transfer time.

A typical deployment scheme of a system for monitoring and estimation of storage performance is shown in Figure 3. The sensors are installed on the relevant storage access nodes. Sensors send data to an aggregating monitoring service. The monitoring data can be displayed via portal or accessed by other components of the data management system, e.g. estimators. The estimators obtain information about the state of a given storage system from the monitoring system or directly from the sensors, depending on the type of estimation used.

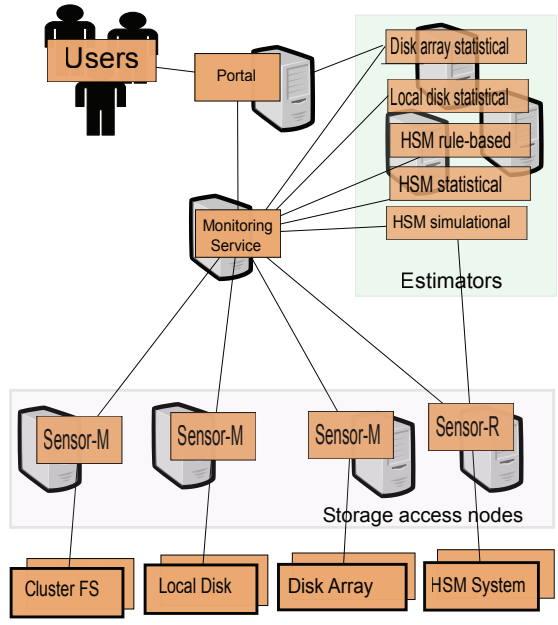


Fig. 3. Storage systems performance monitoring and estimation – a typical deployment scheme

3.3 Ontologies

Modeling of storage systems by using semantic technologies allows for constructing systems on different abstraction levels, basing on semantic interoperability of building components. The differences between the internal construction of storage systems are unified by C2SM model and for monitoring and estimation this is enough for achieving service or system interoperability when exchanging low level parameters. Yet, in order to provide interoperability at higher abstraction levels the usage of semantic based technologies is one of the most convenient ways.

Based on the C2SM model the OntoStor ontology (see Figure 4) has been developed for describing the internal structure of storage resources. The ontology is written in the OWL [24] language. The classes of the C2SM model are represented as OWL classes, while the class properties as data type properties in OWL. In this ontology three relations are used: *is-a*, *hasPart* and *contain*. Below a fragment of the OWL file describing our ATL7100 tape library is listed as an example.

```
<owl:Thing rdf:about="#Tape_Library_ATL7100">
    <rdf:type rdf:resource="#AGH_MediaLibrary"/>
    <AGH_MediaLibrary__NumberOfSlots
      rdf:datatype="&xsd;unsignedByte">
        100
```

```
      </AGH_MediaLibrary__NumberOfSlots>
      <AGH_MediaLibrary__NumberOfDrives
        rdf:datatype="&xsd;unsignedByte">
          4
      </AGH_MediaLibrary__NumberOfDrives>
      <AGH_MediaLibrary__VendorString
        rdf:datatype="&xsd;string">
          ATL
      </AGH_MediaLibrary__VendorString>
      <hasPart rdf:resource="#Drive_Quantum_DLT7000"/>
      <hasPart rdf:resource="#Slot_1_ATL"/>
      <hasPart rdf:resource="#Slot_2_ATL"/>
  </owl:Thing>
```

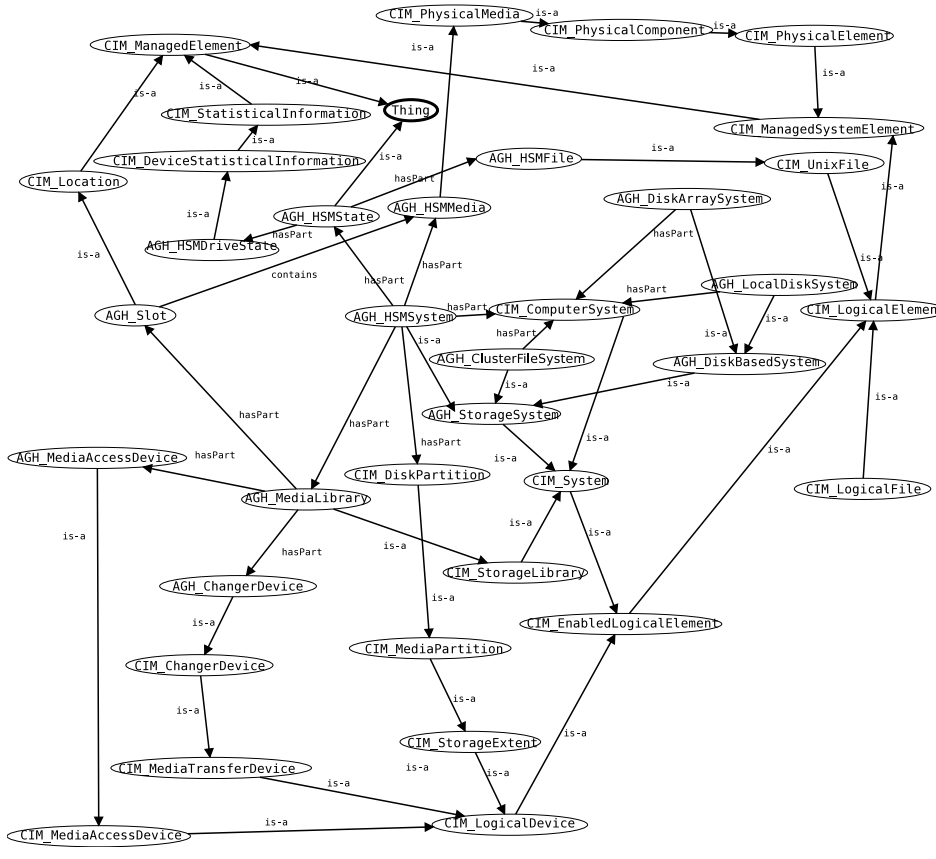


Fig. 4. The OntoStor ontology

In order to allow constructing ontologies for describing QoS and SLA metrics based on low level storage resource performance parameters the OntoStor-ANT ontology has been developed. In this ontology two main concepts are defined: *AGH_Parameters* and *AGH_Attributes*. The performance parameters in this ontology are presented as classes allowing for another type of queries [25]. The OntoStor-ANT ontology can be used in conjunction with upper ontologies, e.g., QoSOnt [26]. For more complex cases, e.g., when multiple ways of calculating a QoS metrics are possible, it is necessary to develop another ontology using the OntoStor-ANT as a base. This is shown in Section 4.3.

4 USE-CASES STUDY

The proposed methodology has been used in a couple of cases for creating a complete system (or fragments of it) for data management with QoS in mind. They are described further beginning with the simplest use case allowing for monitoring the performance of storage systems and ending with a complete solution for accelerating computation of data intensive applications in Grid environment.

4.1 Storage System Performance Monitoring

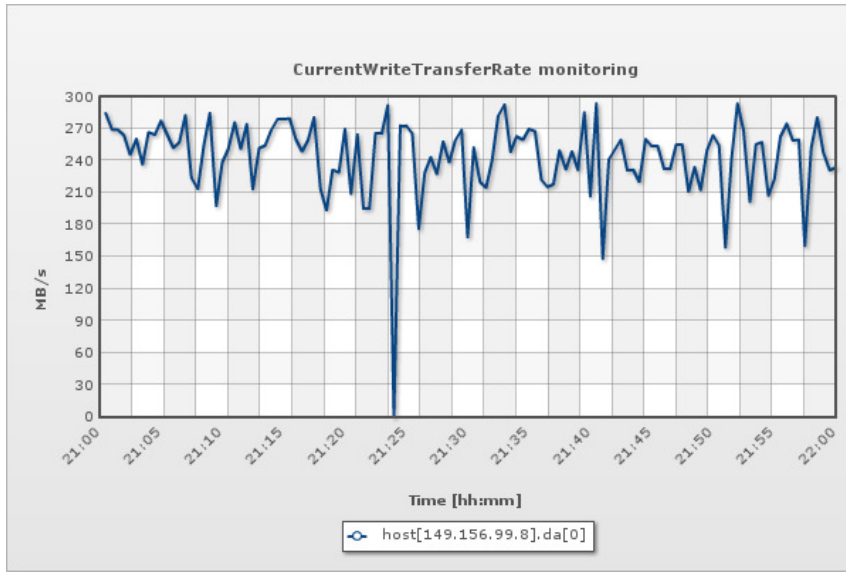
In this use case a monitoring system has been set up consisting of:

- sensors for storage systems (in our case there were: HP FSE HSM storage system, Infortrend disk array and software based RAID on a Linux host),
- monitoring service collecting storage performance related data,
- portal for displaying the monitoring data.

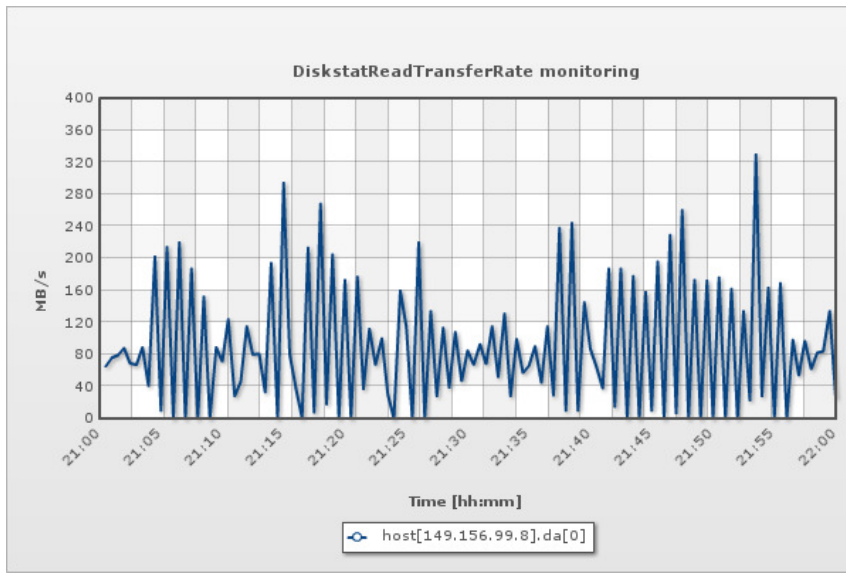
As a monitoring service, the GEMINI2 monitoring system has been used [27, 28]. It has been chosen since it allows for easy plugin of new sensors. Usability tests have been conducted showing that the approach can be used to make conclusions concerning the storage system performance. Selected screenshots for transfer rate monitoring are shown in Figures 5 and 6. We can see that for the disk array an eventual application would be able to write data at average rate of about 250 MB/s for the selected period (see Figure 5 a)) and that there are read transfers going on at average rate of about 80 MB/s (see Figure 5 b)). As for the HSM system performance we can see that an eventual application would be able to read its data at average rate of about 220 MB/s (see Figure 6 a)) and that there are ongoing write transfers at average rate of about 60 MB/s (see Figure 6 b)).

4.2 Data Access Optimization Using Replication Techniques

This use case concerns data access optimization in distributed environments using replication techniques [30, 31]. We assume that the data is physically stored in different storage nodes, which possibly have different performance characteristics

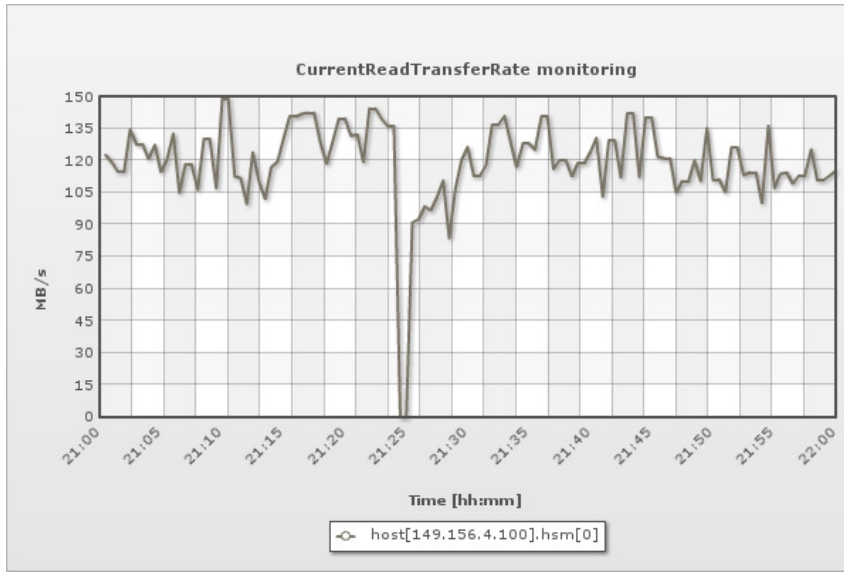


a)

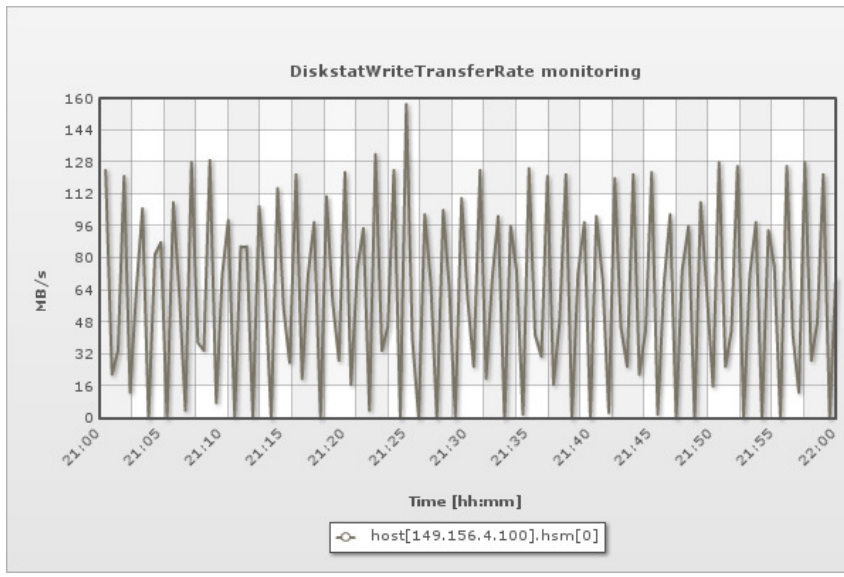


b)

Fig. 5. Disk array performance monitoring; a) Disk array *current write transfer rate*, b) Disk array *diskstats read transfer rate*



a)



b)

Fig. 6. HSM system performance monitoring; a) HSM system *current read transfer rate*, b) HSM system *diskstats write transfer rate*

and have different storage load at a given time. Optimization of data access is done at two points: before the data is written by selecting the best location for creating a new replica or before the data is read by selecting the best replica. In any case we select the location or replica by monitoring the current performance of the storage nodes and based on it predicting (by using the estimators) the future storage performance. An additional component is a replica manager, which takes advantage of the estimation for automatically selecting replicas and locations (see Figure 7).

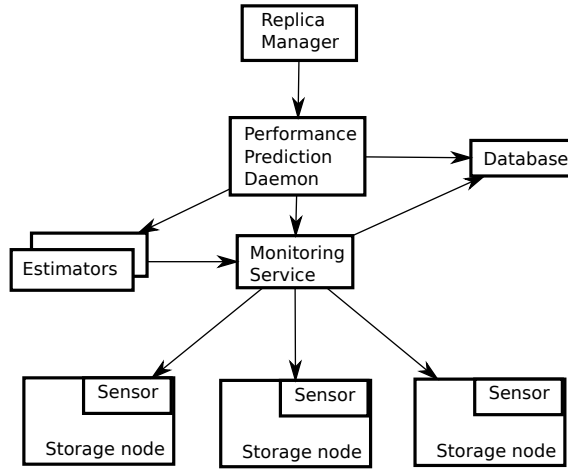


Fig. 7. Replica management in the KMD project

The replica management, including the creation and selection of replicas, is one of the methods of QoS provisioning to the end user application in distributed environments.

This approach has been implemented in the KMD project [32]. The goal of the project is to design and implement a distributed data storage system intended to provide high quality backup, archiving and data access services. In this project replicas are created to increase mainly the data availability and data protection level, but also for increasing the performance of data access. In this case the storage node to be used for the given data transfer is selected based on predefined policies, taking into account different aspects according to the user preferences [33].

4.3 SLA Metrics Monitoring for Storage Services

This use case concerns monitoring of SLA fulfilment regarding storage services. The idea of such system is depicted in Figure 8. The user requirements concerning the storage QoS are specified in the SLA negotiated with the storage service provider. The system notifies the service provider if the SLA is not fulfilled.

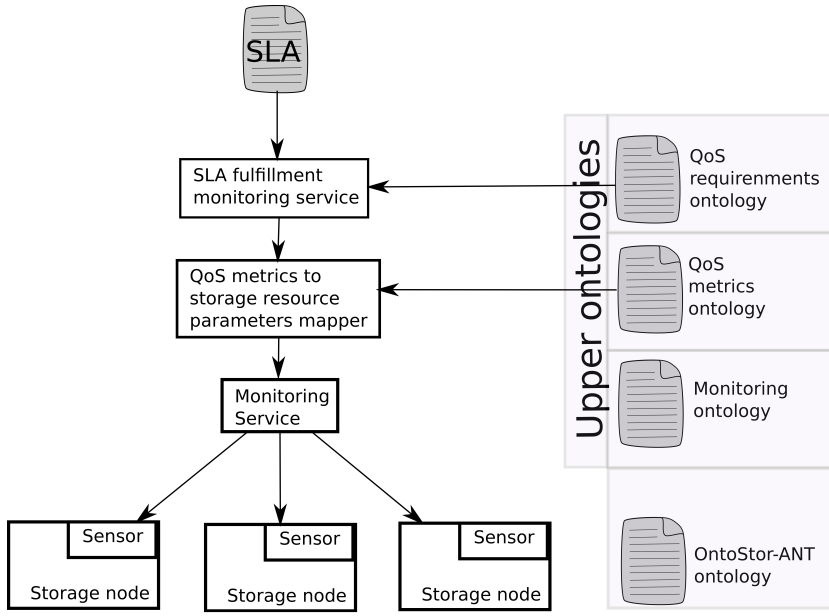


Fig. 8. SLA monitoring system

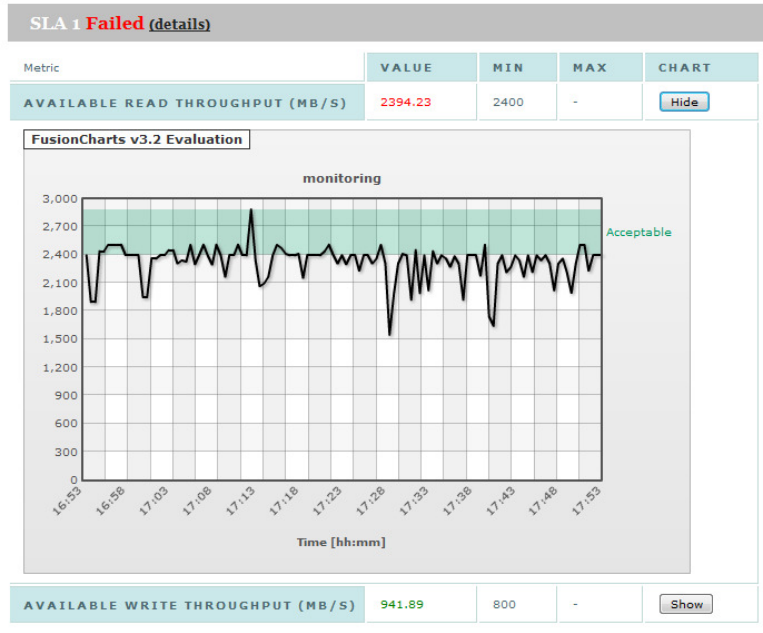


Fig. 9. SLA monitoring

The system consists of: QoS metrics monitoring service, QoS metrics to resource parameters mapper, monitoring service. The system works as follows. The QoS metrics monitoring service specifies, which metrics should be monitored based on QoS requirements ontology and what restrictions are imposed by the SLA. The metrics to be monitored are sent to the mapper, which defines, by using the QoS metrics ontology, a set of low level monitoring parameters needed to calculate the QoS metrics. Next, the mapper obtains the needed parameters from the monitoring system and calculates the QoS metrics, which are then passed up to the QoS metrics monitoring service. The QoS metrics monitoring service checks if the SLA is fulfilled and sends a report to the service provider if needed.

The system has been implemented and deployed on our testing environment. A screenshot showing a chart of the monitored SLA metrics during the last hour is presented in Figure 9.

4.4 Acceleration of Computations for Data Intensive Applications

This use case concerns the speed-up of run-time of applications being executed in a distributed computing environment like grids or clouds and dealing with large amount of data residing on a distributed storage system, like Lustre [29]. The problem is to decide where in the grid the application should be run at the given moment in order to have maximal storage performance. We assume that the storage nodes (or partitions) being parts of the storage system are not of equal storage performance at a given point of time. The diversity of storage performance can be due to: heterogeneity of storage devices in terms of vendor and model; dynamically changing load of devices; decreased performance of devices due to soft (correctable) I/O errors; decreased performance of hard disks due to "sector relocation" because of bad blocks; differences of transfer rates depending on the data placement on the disk plate – inner tracks have lower transfer rates than outer tracks, because of the "zone bit recording" technique implemented in modern hard disk drives.

The system implemented for this use case consists of monitoring system with sensors on the storage and computing nodes and an advisory service, which proposes the most suitable node to be used for running the data intensive application. This approach has been implemented in the QStorMan software [36] developed within the PL-Grid project [34].

The tests showed that a speed-up of up to 40 % compared with running the application without QStorMan can be achieved in our testing environment (see Figure 10). The tests were simulating the storage activity of a data intensive application by running a job, which writes and reads a certain number of big files (128 MB) making pauses between the file transfers to simulate data processing. The chart presents the time of running the job depending on the number of files.

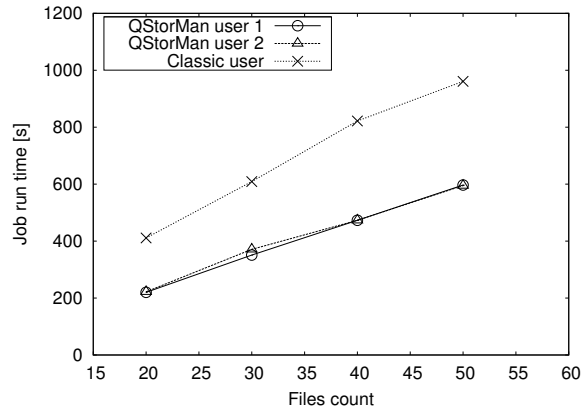


Fig. 10. QStorMan test results for data intensive application (files size is 128 MB)

5 CONCLUSION

With growing demands concerning storage systems and QoS of storage services, it becomes essential to provide monitoring and estimation services concerning the storage system performance. It is not a trivial task because of the complexity and heterogeneity of storage systems. In addition, the load of storage systems changes dynamically, which makes the task even harder. Additional layer in a storage and data management system for uniform presentation of the state of heterogeneous storage system, like the proposed C2SM, is necessary. Thanks to such layer it is possible to achieve service interoperability or even semantic interoperability allowing for creation of other storage resource management layers to cope with the user's requirements concerning storage QoS on top of it. In this way many of the modern distributed environments like clouds and grids could be QoS aware.

In this paper a semantic based storage management methodology has been presented. It could be used for constructing storage QoS related systems. Our research showed that it is possible to manage the storage QoS, but it is necessary to have proper storage performance monitoring of heterogeneous storage systems and the ability to define various QoS metrics and how they are obtained from the lower level MSS parameters. Without using semantic technologies for describing storage systems and services, their usability, flexibility and interoperability are rather limited.

Acknowledgments

This research is supported by Polish MNiSW grant No. N N516 405535. AGH-UST grant No. 11.11.120.865 is also acknowledged. Thanks go to: Renata Słota and Jacek Kitowski for scientific support, Stanisław Polak, Paweł Młocek and Danilo

Lakovic for implementation support, Dariusz Król, Kornel Skałkowski and Michał Orzechowski for helping with testing. This work has been supported in part by Polish Ministry of Science and Higher Education Grant No. 690/N-EGI/2010/0.

REFERENCES

[1] Gantz, J.—Reinsel, D.: The Digital Universe Decade – Are You Ready? May 2010, Available on: http://www.emc.com/digital_universe.

[2] Stewart, G. A.—Cameron, D.—Cowan, G. A.—McCance, G.: Storage and Data Management in EGEE. In: Ljiljana Brankovic, Paul Coddington, John F. Roddick, Chris Steketee, James R. Warren, Andrew Wendelborn (Eds.): Proceedings of the Fifth Australasian Symposium on ACSW Frontiers (ACSW '07), Vol. 68, Australian Computer Society, Inc., Darlinghurst, Australia, pp. 69–77.

[3] The Long Wavelength Array. Available on: http://www.phys.unm.edu/~lwa/, access: 11-03-2011.

[4] Prymula, K.—Piwowar, M.—Kochanczyk, M.—Flis, L.—Malawski, M.—Szepieniec, T.—Evangelista, G.—Minervini, G.—Polticelli, F.—Winiowski, Z.—Saapa, K.—Matczyska, E.—Roterman, I.: In Silico Structural Study of Random Amino Acid Sequence Proteins Not Present in Nature. Chemistry and Biodiversity, Vol. 6, 2009, pp. 2311–2336.

[5] Hluchý, L.—Kussul, N.—Shelestov, A.—Skakun, S.—Kravchenko, O.—Gripich, Y.—Kopp, P.—Lupian, E.: The Data Fusion Grid Infrastructure: Project Objectives and Achievements. Computing and Informatics, Vol. 29, 2010, No. 2, pp. 319–334.

[6] Preston, W. C.: Backup and Recovery. O'Reilly 2007.

[7] Meisner, D.—Gold, B. T.—Wenisch, T. F.: PowerNap: Eliminating Server Idle Power. In Proceedings of ASPLOS 2009, pp. 205–216.

[8] Buccafurri, F.—De Meo, P. et al.: Analysis of QoS in Cooperative Services for Real Time Applications. In: Data Knowl. Eng., Vol. 67, 2008. No. 3, Elsevier, pp. 463–484.

[9] Skital, L.—Janusz, M.—Słota, R.—Kitowski, J.: Service Level Agreement Metrics for Real-Time Application on the Grid. In: R. Wyrzykowski, J. Dongarra, K. Karczewski, J. Wasniewski (Eds.), Proceedings of 7th International Conference PPAM 2007, Gdansk, Poland, September 2007, LNCS 4967, Springer 2008, pp. 798–806.

[10] O'Donoghue, J.—Herbert, J.: A QoS Data Management System within a Pervasive Medical Environment. In: Tom Pfeifer et al. (Eds.): Advances in Pervasive Computing 2006, Adjunct Proceedings of Pervasive 2006, Dublin, 7–10 May 2006, Vol. 207, ISBN 3-85403-207-2, pp. 193–198.

[11] Common Information Model (CIM) Standards, Distributed Management Task Force. Available on: http://www.dmtf.org/standards/cim/, access: 10-03-2011.

[12] Grid Laboratory Uniform Environment (GLUE), GLUE Working Group. Available on: http://forge.gridforum.org/sf/projects/glue-wg, access: 10-03-2011.

[13] Wilkes, J.: Traveling to Rome: QoS Specifications for Automated Storage System Management. In IWQoS (2001), pp. 75–91.

[14] Buyya, R.—Yeo, C.—Venugopal, S.—Broberg, J.—Brandic, I.: Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility. FGCS, Vol. 25, 2009, Elsevier, pp. 599–616.

[15] Litke, A.—Konstanteli, K.—Andronikou, V.—Chatzis, S.—Varvarigou, T.: Managing Service Level Agreement Contracts in OGSA-Based Grids. FGCS, Vol. 24, 2008, pp. 245–258.

[16] Sandeep, U.—Guillermo, A. A.—Gul, A.: Decision QoS: An Adaptive, Self-Evolving QoS Arbitration Module for Storage Systems. Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 04), 2004.

[17] Huang, H.—Andrew, S. G.: Automated Performance Control in a Virtual Distributed Storage System. Proceedings of 9th Grid Computing Conference 2008, pp. 242–249.

[18] Zhang, R.—Chambliss, D.—Pandey, P.—Shearman, W.—Ruiz, J.—Xu, Y.—Hyde, J.: Effective Quality of Service Differentiation for Real-World Storage Systems. In Proc. MASCOTS, 2010, pp. 451–454.

[19] Bigelow, D.—Brandt, S.—Bent, J.—Chen, H. B.: Mahanaxar: Quality of Service Guarantees in High-Bandwidth, Real-Time Streaming Data Storage. In Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST '10), IEEE Computer Society, Washington, DC, USA, pp. 1–11.

[20] Nikolow, D.—Słota, R.—Marmuszewski, J.—Pogoda, M.—Kitowski, J.: Disk Array Performance Estimation. In: M. Bubak, M. Turala, K. Wiatr (Eds.): Proceedings of Cracow GridWorkshop – CGW '09, October 12–14, 2009, ACC-Cyfronet AGH 2010, Krakow, pp. 287–294.

[21] Nikolow, D.—-Słota, R.—Kitowski, J.: Gray Box Based Data Access Time Estimation for Tertiary Storage in Grid Environment. In: Wyrzykowski, R., Dongarra, J., Paprzycki, M., Wasniewski, J. (Eds.): Parallel Processing and Applied Mathematics. 5th International Conference PPAM 2003, Częstochowa, Poland, September 2003, LNCS No. 3019, Springer 2004, pp. 182–188.

[22] Pautasso, C.—Zimmermann, O.—Leymann, F.: RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision. Proceedings of 17th International World Wide Web Conference (WWW 2008), April 2008, Beijing, China, pp. 805–814.

[23] The OntoStor Project web site. Available on: http://www.icsr.agh.edu.pl/ontostor/, access: 10-03-2011.

[24] OWL Web Ontology Language. Available on: http://www.w3.org/TR/owl-features/, access: 10-03-2011.

[25] Polak, S.—Nikolow, D.—Słota, R.—Kitowski, J.: Modeling Storage System Performance for Data Management in Cloud Environment using Ontology. In: S. Pi-

leggi (Ed.), Proceeding of IWSI 2011 International Workshop on Semantic Interoperability. In conjuction with ICAART 2011, Rome, Italy – January 2011, SciTePress 2011, pp. 54–63.

[26] Dobson, G.—Lock, R.—Sommerville, I.: QoSOnt: An Ontology for QoS in Service-Centric Systems. In Software Engineering and Advanced Applications, 31st EUROMICRO Conference 2005, pp. 80–87.

[27] Balis, B.—Kowalewski, B.—Bubak, M.: Real-Time Grid Monitoring Based on Complex Event Processing. Future Generation Computer Systems (2011), DOI 10.1016/j.future.2011.04.005 (to appear).

[28] Balis, B.—Słota, R.—Nikolow, D.—Bubak, M.—Dyk, G.—Lakovic, D.—Kitowski, J.: Using Complex Event Processing for On-Line SLA Monitoring of Data Storage Resources in the Grid. In: M. Bubak, M. Turala, K. Wiatr (Eds.), Proceedings of Cracow GridWorkshop – CGW '10, October 11–13, 2010, ACC-Cyfronet AGH 2011, Krakow, pp. 159–163.

[29] Lustre, `http://www.lustre.org/`.

[30] Słota, R.—Skital, L.—Nikolow, D.—Kitowski, J.: Algorithms for Automatic Data Replication in Grid Environment. In: Roman Wyrzykowski, Jack Dongarra, Norbert Meyer, Jerzy Wasniewski (Eds.): Parallel Processing and Applied Mathematics: 6th International Conference, PPAM 2005, Poznan, Poland, September 11–14, 2005, Revised Selected Papers, Lecture Notes in Computer Science 3911, Springer 2006, pp. 707–714.

[31] Słota, R.—Nikolow, D.—Skital, L.—Kitowski, J.: Implementation of Replication Methods in the Grid Environment. In: P. M. A. Sloot, A. G. Hoekstra, T. Priol, A. Reinefeld, M. Bubak (Eds.): Proc. of Advances in Grid Computing – EGC 2005, European Grid Conference, Amsterdam, The Netherlands, Lecture Notes in Computer Science, No. 3470, Springer 2005, pp. 474–484.

[32] National Data Storage Project, Polish MNiSW grant No. R02 055 03, `https://kmd.pcss.pl`.

[33] Słota, R.—Nikolow, D.—Kuta, M.—Kapanowski, M.—Skalkowski, K.—Pogoda, M.—Kitowski, J.: Replica Management for National Data Storage. In: R. Wyrzykowski, J. Dongarra, K. Karczewski, J. Wasniewski (Eds.): Proceedings of Parallel Processing and Applied Mathematics – PPAM 2009, 8th International Conference, Wroclaw, Poland, September 2009, LNCS 6068, Vol. II, Springer 2010, pp. 184–193.

[34] The PL-Grid Project web site. Available on: `http://www.plgrid.pl/`, access: 10-03-2011.

[35] Majewska, M.—Kryza, B.—Kitowski, J.: On Translation Common Information Model to OWL Ontology. In: M. Bubak and S. Unger (Eds.).: Proceedings of Cracow '06 Grid Workshop, October 15–18, 2006, Cracow, Poland. The Knowledge-based Workow System for Grid Applications, ACC Cyfronet AGH, 2007, pp. 82–89.

[36] Król, D.—Kryza, B.—Skalkowski, K.—Nikolow, D.—Słota, R.—Kitowski, J.: QoS Provisioning for Data-Oriented Applications in PL-GRID. In: M. Bubak, M. Turała, K. Wiatr (Eds.): Proceedings of Cracow Grid Workshop – CGW '10, October 11–13, 2010, ACC-Cyfronet AGH, Krakow 2011, pp. 142–150.

Darin Nikolow obtained his Ph. D. in Poland at the AGH University of Science and Technology in 2003. He is a lecturer at the Department of Computer Science of AGH UST and is specializing in data protection technologies. His reserch interests include storage systems and distributed computing.