Features Management and Middleware of Hybrid Cloud Infrastructures

Evgeny Nikulchev
Moscow Technological Institute,
Moscow, Russia

Evgeniy Pluzhnik
Moscow Technological Institute
Moscow, Russia

Oleg Lukyanchikov
Moscow State Technical University of Radio Engineering, Electronics and Automatics
Moscow, Russia

Dmitry Biryukov
Moscow Technological Institute
Moscow, Russia

***Abstract*—The wide spread of cloud computing has identified the need to develop specialized approaches to the design, management and programming for cloud infrastructures. In the article were reviewed the peculiarities of the hybrid cloud and middleware software development, adaptive to implementing the principles of governance and change in the structure of storing data in clouds. The examples and results of experimental research are presented.**

***Keywords*—Cloud Infrastructure; Distributed Databases; Hybrid Clouds**

I. Introduction

Cloud computing is a paradigm for hosting clusters of data and delivering different services over the network or Internet. Hosting clusters of data allows customers to store and compute a massive amount of data on the cloud. Cloud computing is based on known technology of virtualization of systems in large data centers and the use of existing communication channels.

Currently, capturing and processing big data are related to improving the global economy, science, social affair, education and national security; processing of big data allows us to propose accurate decisions and acquire knowledge from raw data. Several traditional solutions have emerged for dealing with big data such as Supercomputing, Distributed Computing, Parallel Computing, and Grid Computing. However, elastic scalability is important in big data which could be supported by cloud computing services. Cloud computing has several capabilities for supporting big data which are related to handling of big data.

With the growing popularity of cloud services not only the scope and types of services expand (IaaS, PaaS, SaaS), but also the need for new solutions, "cloud" of tasks: providing security and data integrity, quality of service, construction of systems of data flow management and principles for data distribution in the cloud. All of this determined a large amount of scientific research with respect to the clouds that allows us to speak about current complex technologies - cloud technologies (cloud computing technology).

Cloud computing could address two major issues: big data storing and computing. Cloud computing provides a cluster of resources for storage and computing that could be expanded anytime. These features allow cloud computing to become an emerging technology for dealing with big data.

Hybrid architecture at first glance seems to be optimal – it can provide distributed computing and guaranteed security. However, this data network creates a lot of features. These questions are consecrated in the article. In the second part common questions are presented, the third part is devoted to the study and management features. The fourth part describes the development of software applications. In conclusion, there are formulated problems and made general recommendations to overcome them.

The experimental stand with VMWare vCloud was created to simulate the work with hybrid storage. Within this stand it was possible to adapt a software application in dependence on changes in the structure of distributed data storage, as well as test methods for monitoring the operation of applications.

II. Common Problems

From security perspective hybrid cloud is the preferred architecture. As it is known, hybrid cloud deploys data in public and private parts. For safety reasons, we can store in the private parts data that requires secrecy. In the public part - free data access, processing of which requires large computational resources. For example, let's take semistructured data. In that case in public part the main volume is stored, and the codes and algorithms are stored in a private part of the cloud infrastructure.

The public part of the service may be provided by a specialized data center. A private part is an own servers, which can also be a virtual machine (VM).

To provide users with the same features found in commercial public clouds, private/hybrid cloud software must [1]

- provide uniform and homogeneous view of virtualized resources, regardless of the underlying virtualization platform (such as Xen, Kernel-based Virtual Machine (KVM), or VMware);
- manage VM's full life cycle, including setting up networks dynamically for groups of VMs and managing

their storage requirements, such as VM disk image deployment or on-the-fly software environment creation;

- support configurable resource allocation policies to meet the organization's specific goals (high availability, server consolidation to minimize power usage, and so on);
- adapt to organization's changing resource needs, including peaks in which local resources are insufficient, and changing resources, including addition or failure of physical resources.

Although this system has evolved around public clouds — commercial cloud providers that offer publicly accessible remote interface to create and manage virtual machine instances within their proprietary infrastructure — there is also a growing interest in open-source Cloud Computing tools that allow organizations to build their own IaaS clouds using their internal infrastructure [2]. The primary aim of these private cloud deployments is not to sell capacity over the Internet through publicly-accessible interfaces, but to provide local users with flexible and agile private infrastructure to run service workloads within their administrative domain. Private clouds can also support hybrid cloud model by supplementing local infrastructure with computing capacity from an external public cloud.

A hybrid cloud can allow remote access to its resources over the Internet using remote interfaces, such as the web services interfaces used in Amazon [3].

A lot depends on how data structure is decomposed. This is especially noticeable on large data sets. Minor errors in the partitions between the private and the public parts can significantly affect performance of the system. And increasing the number of processors, or other resources won't help. Unlike the paradigm of distributed databases, in a hybrid cloud there are many unknown factors that need to be responded. After all, the hybrid cloud is the infrastructure, there are two independent cloud storages with their own management systems. Compounding and switching systems is an unknown factor as well as information delivery routes between clients and parts of the system. In the absence of control, application may spend most of the time waiting for the response, rather than computing tasks of data processing. These situations should not only be addressed in time but are under constant monitoring. It is necessary to determine the point at which to start the mechanism of redistribution of resources in the network. All of the above applies primarily to large data. The experiments with small amounts of data and small queries are not sensitive to it. And this refers specifically to a hybrid cloud that constantly exchange data between private and public parts.

For designing information systems in the cloud, there are the following problems:

- Inability to assess the execution of individual requests and the flow request.
- There are no general principles of designing systems with large amounts of data (BigData).
- A considerable amount of data is semistructured (XML).
- No migration technologies to the cloud, hence the need to rewrite code when porting.
- There are no generally accepted principles for virtualization management and resource allocation in the cloud.
- Limitations associated with the use of data communication protocols.

The task was to study these features and develop the technology to create applications in a hybrid cloud. Within the framework of these limitations, principles were formulated to develop methods that provide guaranteed quality of functioning of the application.

- Design of the systems should be based on preliminary study, on simulation and experimental models.
- It is necessary to control basic parameters of the infrastructure.
- The use of object-oriented design technologies for database modifications.
- Technological systems should provide the flexibility to change structure, volume of data, the number of requests.

III. Hybrid Cloud Management

Relative performance differentiation schemes have been identified as promising approaches to achieve the performance differentiation objectives of multiple classes workloads and applications. The main challenge of applying it in a shared resource environment is to maintain the above performance differentiation ratio by using dynamic resource allocation under varying resource demands and workloads [4]. The system controlled by the controller is referred to as the target system. The target software system provides a set of performance metrics for properties of interest referred to as outputs. The sensors monitor outputs of the target system, while controlling inputs. The controller is the decision making unit of the control system. Its main objective is to maintain the outputs of the system sufficiently close to the target values, by adjusting the control inputs. These target values are called the set point signals. This gives the option for the designer to specify the goal/desired values for the outputs.

The control system design generally consists of two main steps. First, a formal relationship between the control inputs and outputs has to be constructed. In control theory this relationship is referred to as the dynamic model of the system. System identification is typically used to construct the system model using the measurements of input and output data. This model of the system is then utilized in the subsequent steps which include controller design, simulation, analysis and testing using well established tools in control engineering.

Many control engineering approaches have been proposed for data center management in the past few years. Works in [5], [6], [7] have proposed different control techniques to manage power and performance properties at absolute value. Several techniques control response time by manipulating CPU utilization in virtualized environments.

The relative performance automatic control with feedback have been utilized to manage web servers, storage systems and data centers . In [8] were investigated three different ways to formulate the input and output variables of the relative management scheme in order to reduce the nonlinearity, while maintaining the scalability and applicability of the feedback control. Final results indicated that taking the ratios of the response time and resource caps of consecutive client classes based on the priority is the most suitable and effective setting.

To develop applications with the distribution of big data in the hybrid cloud, the experimental stand can be used. Experience shows that for each system it is required to check the load and response time, generate popular searches [9]. The experiments allow to find out the nonlinearity in each particular system.

There was created an experimental stand (ES) that simulates the work with hybrid storage. Stand itself and some experimental results obtained with it are described in [10], see fig. 1. Deployed software VMWare vCloud allows organization at all levels. VMware ESXi is used on two servers to create a cloud in the ES. Management system VCenter and application VMware vCloud Director are deployed. In ES there are more than 15 physical Cisco 29 switches and routers Series 26 and Series 28, as well as virtual switches Nexus. System based on ES allows simulating routes of access to data, converging and diverging channels (can be done dynamically).

As a criterion for the efficiency of the data structure it is proposed to use structure decomposition between distributed repositories, in which secure access to the data users will be provided. A criterion is given in the form of a range of data delivery time for test requests. Clearly, this criterion is strongly dependent on many parameters which are specific for each case in each system, namely:

- ─ actual distance to the end user (or data delivery route);
- ─ load of the network;
- ─ current number of users and the complexity of their application request;
- ─ downloaded applications, ensuring data collection in the private and public section;
- ─ actual loading of resources in the cloud.

Many are nonlinear and cannot be calculated in fully automatic mode for the general case [11]. To configure and operate instructions, it should be possible to implement monitoring system (and preferably BPCS system to monitor and control loading). Example of operational monitoring parameters is shown (Fig. 2).

A series of experiments with different demands and workload simulation channels (Fig. 3) shows a portion of the network traffic.

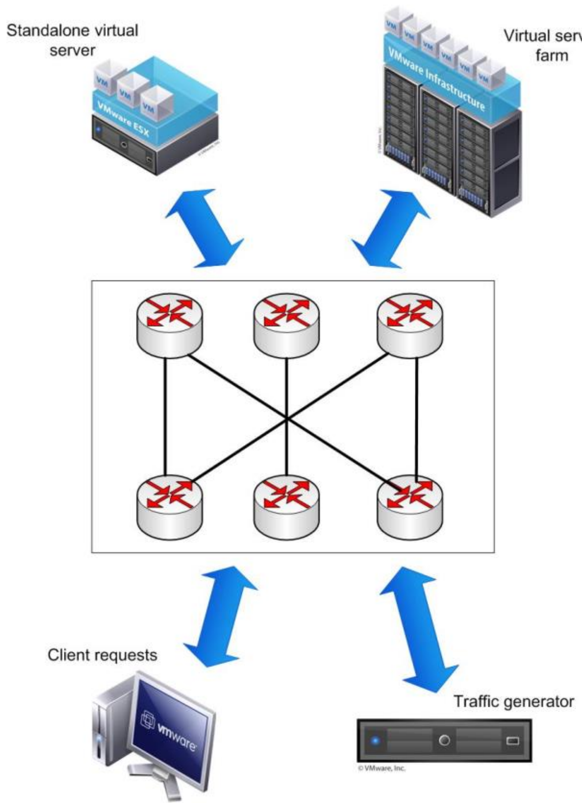
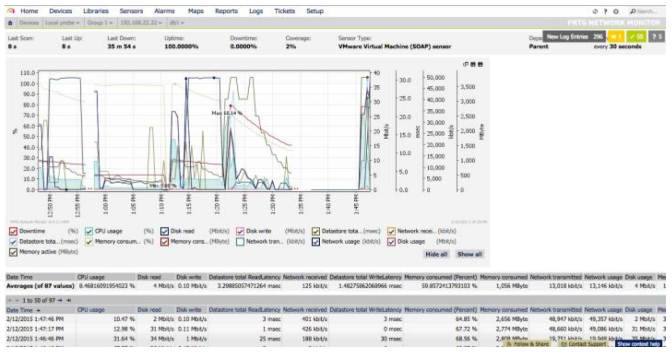


Fig. 1. Experimental installation

However, all these features need to be considered during development of applications. This is done by monitoring systems with feedbacks and policy management. In case of fully loaded system or one of the elements of the infrastructure (network, storage in a private cloud), it is possible not to give the implementation of requests of new users, allowing to perform the previous ones. There are a number of parameters that can be calculated during the design phase. Although it is impossible to evaluate the effectiveness of the algorithm for the cloud infrastructure for data distribution and virtual computing resources (due to the scale, it is impossible to assess in advance how much will be allocated to the algorithm), but you can choose the most popular complex queries, and define the data structure for them. In this case the execution of these requests will be guaranteed to get at a given level of service quality.



Last Scan:	Last Up:	Last Down:	Uptime:	Downtime:	Coverage:	Sensor Type:	Dependency:
8 s	8 s	35 m 54 s	100.0000%	0.0000%	2%	VMware Virtual Machine (SOAP) sensor	Parent

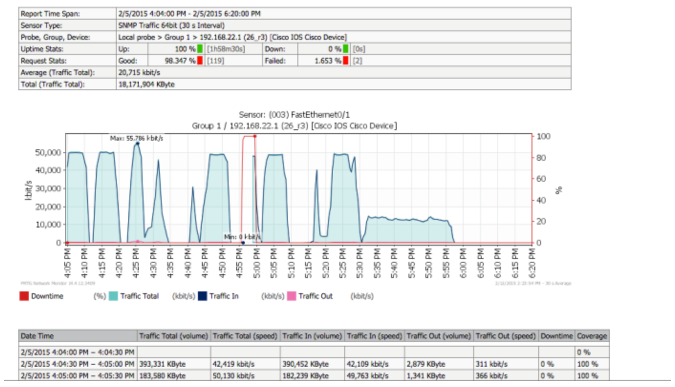
every 30 seconds

Date Time	CPU usage	Disk read	Disk write	Datastore total ReadLatency	Network received	Datastore total WriteLatency	Memory consumed (Percent)	Memory consumed	Network transmitted	Network usage	Disk usage
Averages (of 87 values)	8.46816091954023 %	4 Mbit/s	0.10 Mbit/s	3.29885057471264 msec	125 kbit/s	1.48275862068966 msec	59.8572413793103 %	1,056 MByte	13,018 kbit/s	13,146 kbit/s	4 Mbit/s

1 to 50 of 97

Date Time	CPU usage	Disk read	Disk write	Datastore total ReadLatency	Network received	Datastore total WriteLatency	Memory consumed (Percent)	Memory consumed	Network transmitted	Network usage	Disk usage
2/12/2015 1:47:46 PM	10.47 %	2 Mbit/s	0.10 Mbit/s	3 msec	401 kbit/s	3 msec	64.85 %	2,656 MByte	48,947 kbit/s	49,357 kbit/s	2 Mbit/s
2/12/2015 1:47:17 PM	12.98 %	31 Mbit/s	0.11 Mbit/s	1 msec	426 kbit/s	0 msec	67.72 %	2,774 MByte	48,660 kbit/s	49,086 kbit/s	31 Mbit/s
2/12/2015 1:46:46 PM	31.64 %	34 Mbit/s	1 Mbit/s	25 msec	188 kbit/s	30 msec	68.56 %	[illegible]	[illegible]	[illegible]	[illegible]

Fig. 2. Type of system for monitoring parameters of hybrid clouds in the experimental stand



Report Time Span:	2/5/2015 4:04:00 PM - 2/5/2015 6:20:00 PM					
Sensor Type:	SNMP Traffic 64bit (30 s Interval)					
Probe, Group, Device:	Local probe > Group 1 > 192.168.22.1 (26_r3) [Cisco IOS Cisco Device]					
Uptime Stats:	Up:	100 % 🟩	[1h58m30s]	Down:	0 % 🟩	[0s]
Request Stats:	Good:	98.347 % 🟥	[119]	Failed:	1.653 % 🟥	[2]
Average (Traffic Total):	20,715 kbit/s					
Total (Traffic Total):	18,171,904 KByte					

Date Time	Traffic Total (volume)	Traffic Total (speed)	Traffic In (volume)	Traffic In (speed)	Traffic Out (volume)	Traffic Out (speed)	Downtime	Coverage
2/5/2015 4:04:00 PM – 4:04:30 PM								0 %
2/5/2015 4:04:30 PM – 4:05:00 PM	393,331 KByte	42,419 kbit/s	390,452 KByte	42,109 kbit/s	2,879 KByte	311 kbit/s	0 %	100 %
2/5/2015 4:05:00 PM – 4:05:30 PM	183,580 KByte	50,130 kbit/s	182,239 KByte	49,763 kbit/s	1,341 KByte	366 kbit/s	0 %	100 %

Fig. 3. Traffic Analysis

IV. Middleware Features

Development of the information systems and database applications led to the emergence of technologies such as Open Database Connectivity (ODBC), Data Access Object (DAO), Borland Database Engine (BDE), providing a common programming interface for working with various databases. In the future, the needs of the development of software and hardware that work with the data required to ensure access to non-SQL data stores, e-mail and directory services. To provide these functions new technologies have been developed — Object Linking and Embedding, Database (OLEDB) and ActiveX Data Objects (ADO). With the advent of powerful class systems Frameworks, such as .Net and Qt, data processing technologies became embedded into the database, providing full integration with them, as well as integration with semistructured data in XML. The latter became a common format for storing data in files. Efficient technology to communicate with relational data objects is object–relational mapping ORM [12].

Using middleware ORM is of great use in such implementations as QxORM, EntityFramework, Dapper, Hibernate, and others. But all of these technologies are effectively used for the data stored and managed in only one database. Using ORM technology automates location control for data. With classic design, the designer must be sure to specify data location in each request for a hybrid cloud software to connect and disconnect from the database. All this leads to an increase in complexity of software development and causes errors in the code. ORM allows to incorporate the essence of each attribute responsible for the physical location of data in a distributed system. The development of relevant ORM technology for a hybrid cloud infrastructure is the development of intelligent module that determines the optimum storage of data to improve the performance of the system. The optimality of data storage should be automatically determined based on many criteria, such as network bandwidth, server load, number of customers and others. Many of these parameters can be obtained experimentally. Therefore, intelligent control module must adapt based on the information collected from all storage systems during trial operation.

Big Data is not limited in the use of relational tables, semistructured data, data files, etc. ORM requires modification to be used with heterogeneous data: while preserving the basic functionality it is needed to allow the programmer to operate with object classes.

It is necessary to come up with technology for development that would manipulate the data in the hybrid cloud providing following functions

1) *The basic operations for interaction with the objects:*
- - Select;
- - Update;
- - Insert ;
- - Delete.

2) *The presence of the transaction control:*
- - Commit the transaction (commit);
- - Transaction rollback (rollback);

3) *Storage of information about distant objects ("map"):*
- - Registration of the object (registration);
- - Cancellation of registration (unregistration);

Architecture prototype «ArPlatform», which is a middleware service for developing an application is shown in Fig. 4. The prototype includes service «ArNotifyService» Library and «ArLib». There are implemented classes in which you can specify the location and distribution of data, classes that provide functionality for management at the program level.

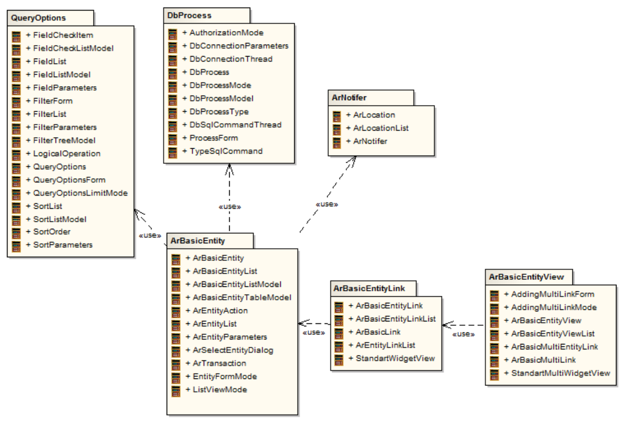


Fig. 4. Structure «ArPlatform»

Examples of operating with the service:

```
Student * student  = new Student();
student->setId(5);
DbConnectionParameters db1;
db1.setDbConnectionFile(QDir::homePath()
+QDir::separator()+"connection.ini");
student->selectData(db1);
student->insertData(
            ArEntityNotifyReceiver::instance()->
            dbEntityMap().items().filtredByAppName("serviceSt
udents"));
Student * updateStudent = new Student();
updateStudent->setFio("testcommit");
```

updateStudent->setMark(4.3);
student->updateData(updateStudent,
ArEntityNotifyReceiver::instance()->
dbEntityMap().items().
filtredByHost(QHostAddress(192.168.0.3));

In this example data may be stored not only in relational databases but also in files, semistructured instances.

Data manipulation with the network structure:
DbConnectionParameters db1;
db1.setDbConnectionFile("connection.ini");
student->updateData(updateStudent,db1);

Data service manipulation during installation on a node in a private cloud

student->updateData(updateStudent,
ArEntityNotifyReceiver::instance()->
dbEntityMap().items().
filtredByAppName("serviceStudents"));

The aim of the experiment is to determine the efficiency of separation of the DB into 2 parts: public and private. For the experiment 3 compute nodes were prepared, the overall structure of which is shown in fig. 5:

Public DBMS server, which is more powerful .
Private DBMS server, which is less powerful.

A client that makes requests to the published server using specialized software.

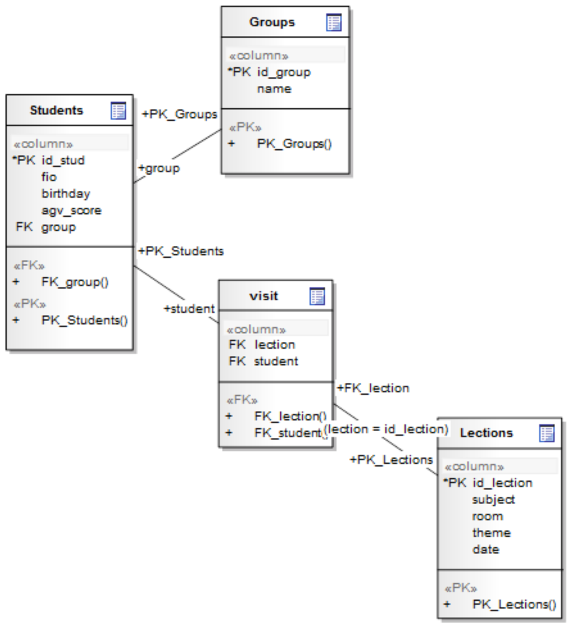


Fig. 5. The database scheme

Part of the database included in the educational process at the University was used for experiments (fig. 6).

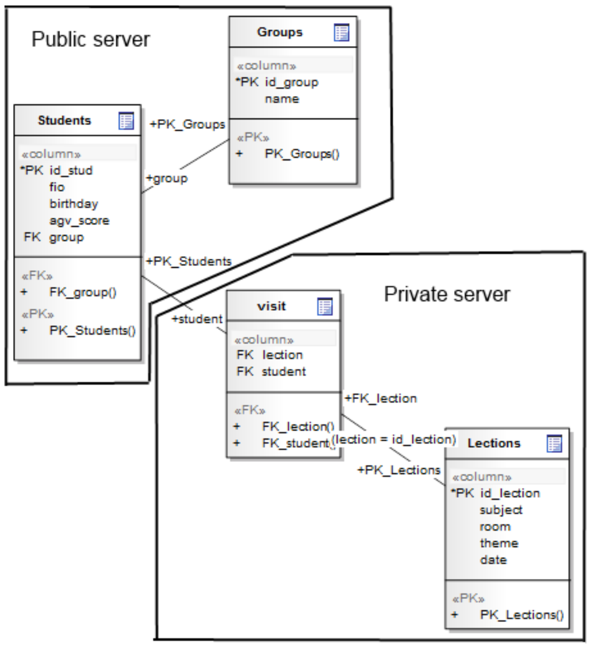


Fig. 6. Split the database between the public and private servers

Table "Students" contains the following information about students:

- "id_stud" - unique number of the student, it is the primary key;
- "fio" - surname, name and second name of the student;
- "birthday" - the date of student's birth;
- "agv_score" - the average score;
- "group" - the group number, which includes the student; is an external key of the table "Groups".
- Table "Groups" associated one-to-many with table "Students" contains the following information about groups:
- "id_group" - unique group number;
- "name" - the name of the group.

Table "Lections" associated many-to-many with table "Students" via table "visit", contains the following information about the lectures:

- "id_lection" - unique number of lectures;
- "subject" - the number of the held object;
- "room" - the cabinet number (the audience);
- "theme" - is the theme of the lecture;
- "date" - the date of the lecture.

- Table "visit" includes the attendance of students.
- "id_student" - the number of student. Is external to the key of the table "Students";
- "id_lection" – the number of the lecture. Is external key to the table "Lections".

The results of the query fetching all data from table "students" (adding data from tables associated one-to-many and many-to-many) were compared to test the effectiveness of the separation of the database into public and private parts.

In the first case, all of the database was on a public server, to retrieve the data following query was used:

```
select * from students.students
left join students.groups on students."group" =
groups.id_group
left join (select * from students.visit
left join students.lections on visit.id_lection =
lections.id_lection) t1
on students.id_stud = t1.id_student.
```

In the second case the part that relates many-to-many with table "Students" was placed on a private server (figure 4). Data was obtained by the function PostgreSQL dblink, which allows to perform the query to another DBMS. Request in the second case:

```
select * from students.students
left join students.groups on students."group" =
groups.id_group
left join (select * from dblink('hostaddr=xxx.xxx.xxx.xxx
port=xxxx dbname=… user=… password=…', 'select
id_student,lections.id_lection,id_subject,date,room,theme
from students.visit
left join students.lections on visit.id_lection =
lections.id_lection') as t(id_student INTEGER,id_lection
INTEGER,id_subject INTEGER,date DATE,room
INTEGER,theme TEXT)) as t1
on students.id_stud = t1.id_student
```

Using such functions as dblink (which allows to perform queries to another DBMS), makes it possible to bring part of the database to another DBMS with changes to the query without need to modify client application.

The experimental results shown in Fig. 7 and 8.

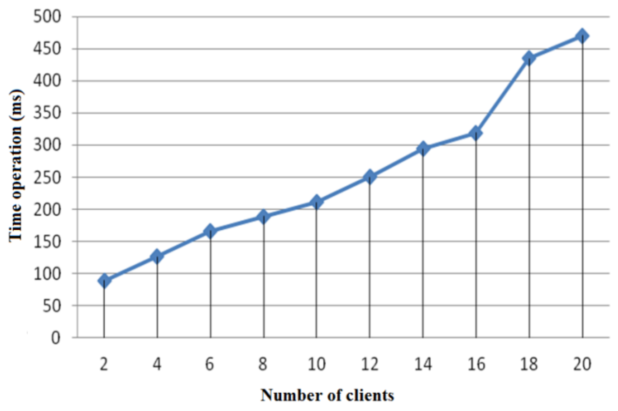


Fig. 7. Experimental query without using ORM technology

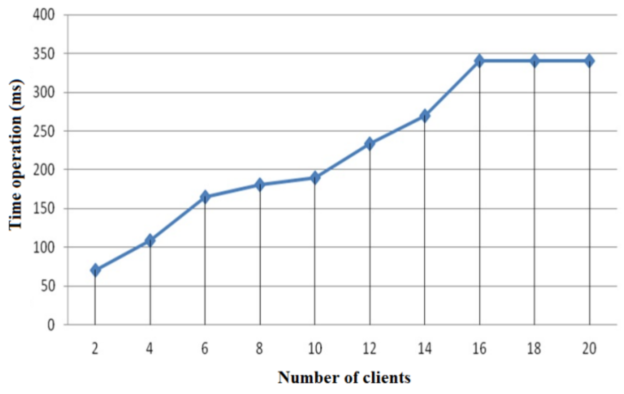


Fig. 8. Experimental query using development

It should be noted that when using a class developed with an increase in the number of clients during the operation grows

V. CONCLUSION

Main feature of the application is an intermediate layer that implements the connection of user requests to the location of distributed data. Presence of unknown destination switching when using public cloud and mobile client makes it impossible to estimate the time of the algorithms. That is why you want to use software technology to control all stages of the system. The hybrid infrastructure has many positive aspects of cloud computing: scalability, virtualization and also (due to the distribution of data) safety and security of data.

The task was to account these features and develop the technology that creates applications in a hybrid cloud.

The principles of development were formulated to provide guaranteed quality and functioning of the application.

1) The system design should be based on a preliminary study on the simulation and experimental models.

2) It is necessary to control the main parameters of the infrastructure.

3) The use of object-oriented technology modifications of database design.

4) Technology should provide the flexibility of system structure, data volume, number of requests.

The result shows that in the second case, when the database is divided, the average query execution time is much higher than with solid database. Therefore the separation of the database on the public and private parts adversely affects the performance of the system, and only has advantages from the viewpoint of safety. Distributed data complicates the development of the software, making it difficult and time-consuming to use common programming techniques. Despite the development of technologies such as .Net and Qt, developers eventually have to operate SQL queries and clearly prescribe the access to distributed data. In the context of widespread object-oriented development methodology and application systems, with relational DBMS having dominant position in the market, attractive solution is to use intermediate software that provides necessary object-oriented interface to data stored under the control of a relational

DBMS. To communicate with developed relational data objects there was used special technology. The essence of this technology is in accordance of programming entity to relational database object: each field of a table is assigned to a class attribute of the object.

The basic steps are the following:

1) Determination of the basic structure of the physical distribution of data in the hybrid cloud.

2) Development of database structure.

3) Development of methods for data processing based on physical location of the data.

4) Creating classes of objects, including data and methods for their treatment.

5) Modification of the structure as a result of an experimental study on the simulation bench.

6) Changing methods of processing inheritance

Depending on the task, system can be restructured to increase the speed of the most common queries, or to perform the most demanding requests in the public cloud. Formulated features and developed design technology for middleware will allow applications for hybrid clouds to be effectively designed. This provides the possibility to adapt a software application in dependence on changes in the structure of distributed data storage in clouds, gives methods for monitoring the operation of applications and management principles.

REFERENCES

[1] M. Bahrami, M. Singhal, "The role of cloud computing architecture in big data," Information Granularity, Big Data, and Computational Intelligence, Vol. 8, pp. 275-295, 2015.

[2] B. Sotomayor, R. S. Montero, I. M. Llorente, I. Foster, "Virtual infrastructure management in private and hybrid clouds," IEEE Internet computing, Vol. 13, pp. 14-22, 2009.

[3] S. Borja, M. S. Ruben, M. T. Ignacio, "An Open Source Solution for Virtual Infrastructure Management in Private and Hybrid Clouds," IEEE Internet Computing, Vol. 1, pp. 14-22, 2009.

[4] M. Bahrami, "Cloud Template, a Big Data Solution," Journal of Soft Computing and Software Engineering, Vol. 3, n. 2, pp.13-17, 2013.

[5] C. Lu, Y. Lu, T. Abdelzaher, J. Stankovic, S. Son, "Feedback control architecture and design methodology for service delay guarantees in web servers," IEEE Transactions on Parallel and Distributed Systems, Vol. 17, n. 9, pp. 1014-1027, 2006.

[6] D. Kusic, N. Kandasamy, G. Jiang, "Combined power and performance management of virtualized computing environments serving session-based workloads," IEEE Transactions on Network and Service Management," Vol. 8, n. 3, pp. 245-258, 2009.

[7] X. Wang, Y. Wang, "Coordinating power control and performance management for virtualized server clusters," IEEE Transactions o Parallel and Distributed Systems, Vol. 22, n. 2, pp. 245 –259, 2011.

[8] X. Wang, M. Chen, X. Fu, "MIMO power control for highdensity servers in an enclosure," IEEE Transactions on Parallel and Distributed Systems, Vol. 21, n. 10, pp. 1412 –1426, 2010.

[9] Y. Lu, T. Abdelzaher, C. Lu, L. Sha, X. Liu, "Feedback control with queueing-theoretic prediction for relative delay guarantees in web servers," Proceedings 9th IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 208 –217, 2003.

[10] E. Pluzhnik, E. Nikulchev, S. Payain, "Optimal control of applications for hybrid cloud services," 2014 IEEE World Congress on Services, pp. 458-461, 2014. doi: 10.1109/SERVICES.2014.88

[11] E. Nikulchev, E. Pluzhnik, D. Biryukov, O. Lukyanchikov, "Experimental Study of the Cloud Architecture Selection for Effective Big Data Processing," International Journal of Advanced Computer Science and Applications, Vol. 6, n. 6, pp. 22-26, 2015.

[12] T. Patikirikorala, L. Wang, A. Colman, J. Han, "Differentiated Performance Management in Virtualized Environments Using Nonlinear Control, IEEE Transactions on Network and Service Management," Vol. 12, n. 1, pp. 101–113, 2015.

[13] O. Lukyanchikov, E. Pluzhnik, S. Payain, E. Nikulchev, "Using object-relational mapping to create the distributed databases in a hybrid cloud infrastructure," International Journal of Advanced Computer Science and Applications, Vol. 5, n. 12, pp. 61-64, 2014.