6th International Conference on Strategies, Models and Technologies of Economic Systems Management (SMTESM 2019)

# Information System "Electronic University": the Experience of Khmelnytskyi National University

Andrii Mazarchuk
*Khmelnytskyi National University*
*Department of Automated Systems and Modeling in Economics*
Khmelnytskyi, Ukraine
a.mazarchuk@gmail.com

Constantin Belovsky
*Khmelnytskyi National University*
*Department of Automated Systems and Modeling in Economics*
Khmelnytskyi, Ukraine
constantin.belovsky@gmail.com

Tetiana Zavgorodnia
*Khmelnytskyi National University*
*Department Of Automated Systems And Modeling In Economics*
Khmelnytskyi, Ukraine
zavgorodnyatp@gmail.com

***Abstract* — The article is devoted to the development of a university management information system based on the experience of developing and implementing the Electronic University system at the Khmelnytskyi National University. We consider the main tasks solved by the university management system, taken architectural decisions, the principles of an effective database structure design, development tools, report generation, programming languages and frameworks. A comparative analysis of possible approaches is carried out and recommendations are made taking into account practical experience in using the system, which has been developed and used since 2001. The main business processes and functional structure of the information system are considered. The need and possibility of storing data that reflect the history of all changes are justified. This provides the possibility of obtaining a data slice or the state of a certain information object at any time by certain calculations in the database. It also gives you the opportunity to get almost any statistics for any interval or at any time.**

***Keywords — Information system, database, design, development.***

## I. INTRODUCTION

Most universities have a limited budget for the development and implementation of information systems. Commercial solutions of well-known large companies, such as the Oracle, can cost about $1.5 - $2 million, and subsequent support cost can reach $500,000 a year. Such projects are viable within the grant funding, and for most universities unaffordable. Even domestic development can cost about hundreds of thousands of hryvnias and tens of thousands of hryvnias a year to support, which most universities are not willing to pay.

Under these conditions, the development of an information system by the university itself, with appropriate specialists, is quite viable. With a good initial planning of the structure of an information system, it can gradually develop, adding new functions and capabilities without the need to constantly rework the entire system from scratch. Such an approach will not give a quick result, however, the introduction of large “ready-made” information systems, which is often accompanied by partial or deep adaptation of the modules of the system to the specific needs of the university, can take years, and the inconsistency of the structure of such a system with the changing business processes of the university after several years of implementation and significant costs may generally lead to the abandonment of the use of such a system. Therefore, a departure from the universal type of the system and the development of its own functional modules in practice can give even more significant and faster results, and specific modules without any adaptation will work in strict accordance with the requirements of the customer. Adaptation of modules in case of business process changes at the university can also be performed by its own developers.

Strictly speaking, the development of an information system is a process of continuous development. After the implementation of the initially ordered modules, there are always new requirements, both for the development of existing modules and the development of new ones for various departments or tasks. Therefore, the choice of development tools / languages is very important. The main requirement is openness. With a sufficiently long development period, it is very important to be able to introduce new technologies, which could simply not exist at the time the project started, and for this to happen, there must be the possibility of integrating existing ones with new ones. Binding to closed commercial solutions in the future often leads to problems. And given the limited budget, the most suitable are Open Source solutions. It should also be considered that the use of open standards makes it easier to find developers with the appropriate qualifications, which is important with the long-term development of the project.

## II. SELECTION OF DEVELOPMENT TOOLS

Our information system "Electronic University" began to be projected from 1999, when most developers used native binary code applications obtained using compiled programming languages. We proceeded from the requirements of the work of users with different operating systems in a distributed network, both within the university and beyond. Updating of versions of modules should occur often and quickly. Therefore, we decided to design the system using a client-server technology using web interfaces [1]. This approach was fully justified not only then, but also showed promise for many years to come, when the rapid development of web technologies provided the ability to build applications of any level of complexity, both for personal computers and for tablets or smartphones.

The main advantage of using web technologies is the possibility of centralized development and support of the system when users can access it from anywhere, both inside the university, and from home or on the road. At the same time, it remains possible to restrict access to critical information only from specific devices / subnetworks during authorization, both with a password and / or security certificates (most often such requirements are put forward to access financial information). Some features of the work of users with confidential information through the web interfaces have set appropriate requirements for the means of security. For example, the connection between the client and the server should have been encrypted, user authentication should be provided by a login/password pair, some key elements transmitted by web interfaces must be digitally signed, sql queries to the database must use parameterization.

PHP was chosen as the main programming language for the server side for its simplicity and ease of use. This choice was influenced by the fact that it was used to create more than half of the sites on the Internet. And this, in turn, determined a huge number of developers using PHP, as well as tools, frameworks, documentation and support. Although many corporate systems are developed using Java technologies, practical experience in developing websites using them, as well as interaction with existing solutions, shows that, as a rule, the development process using Java is much more resource intensive than PHP, and the production version in Java work significantly slower than PHP. We do not claim that Java projects are necessarily slower than PHP, but Java is a more universal and complex platform that requires high qualifications for effective use. The use of PHP for developing web applications even for less skilled developers, gives the opportunity to develop high performance websites. In other words with a team of qualified Java developers, Java is an excellent choice, but PHP will allow you to get almost the same results, with the requirements for developers can be lower and find such developers much easier.

Since most popular modern frameworks simply didn’t exist at the time of the development of the Electronic University, we developed our own framework based on the use of templates, allowing us to quickly and compactly describe a standardized web interface built on “cascade” tables that often follow the structure of the database, which allowed to significantly reduce development time.

By “cascade” tables we understand tables where, in addition to data, buttons are displayed on the table row for show related records from other tables. Those tables may also have related tables, etc. When you click on the button, the current table is collapsed, leaving only the selected row with header and the selected linked table is displayed with a small indent under the row. Thus, opening the following levels of nesting, you get one row (with a header) of each table along the path and in full (the current page) of the last selected linked sub-table. Visually, it looks like a cascade of tables, where you can see the data from all related records of different tables that led to the current displayed table. At the same time, clicking on the heading of the corresponding table allows you to immediately return to this table, bypassing the intermediate steps, similar to breadcrumb in modern file managers, this feature has been implemented as the basis of the interface of the Electronic University since 2002.

This approach allows a unified description of the table display structure (a table here also means the result of a complex query) and simplifies rendering. With the traditional approach, when all the details of the selected row of the table are displayed in a separate dialog box, placing information from a variety of related tables in it at the same time becomes problematic and leads to either oversaturation of information on the screen or requires significant additional time spent on optimizing the placement of all data in a user friendly manner. In this case, as a rule, the transition to the next level of detail hides data from previous levels, complicating the perception of the data chain by which you got to this point and it is impossible to return to an arbitrary intermediate level. Thus, cascading tables can significantly reduce the development time of a complex information system in which a large amount of interconnected data is used.

It should also be noted that for each table in the cascading chain, you can determine the parameters that will be calculated based on the selected record. These parameters are available during the formation of related tables and can be used both in a query to form a modified set of records, and when forming an available set of actions for each table record. Also, each table can have not only one form for editing and one for adding records, but a whole set of forms, the choice of which depends on the parameters generated at previous levels. For example, the “Orders” table may have the “List” sub-table for viewing the general list and the “Order categories”, which in turn may have the “Students” sub-table. When get to the “Students” table from a certain category of order, columns set is defined by the category, and the form for adding and editing a record also selected by category. For example, in categories with a change in the financial form of training, it is displayed in the table and it can be changed in the form for editing, in the category for transfer from group to group, there can be only the names of groups in the table and financial form change shouldn’t be allowed when editing (the field can simply be absent on the form). Such a declarative approach to the description of the interface allows in many cases to not use programming in its pure form, which also significantly speeds up development and reduces the number of potential errors.

Initially, the client interface of the system was built using standard HTML with minimal use of JavaScript. This approach imposes minimal requirements on client computers, which is especially important for universities, where the hardware is usually updated infrequently and it is quite possible to meet quite old computers with low productivity.

Today, there is a situation where even the performance of outdated computers 5-7 years old is enough to run applications that are completely built with HTML5, and which give the client application substantially greater opportunities comparable to native applications. Therefore, it is advisable to create new projects / modules based on this technology and its derivatives (Progressive App and so on). That, however, does not exclude the simultaneous use of modules built with “old” technologies.

In this case, the role of the client application is reduced to working with data, and there is almost no need to create an interface on the server side. When choosing a modern PHP framework, you can recommend Symphony and Laravel. The choice between them depends on the chosen role of PHP in the formation of the interface. If basically it is intended to provide a fast and efficient REST interface to ensure interaction with the Rich Client, then the optimal choice will be Symphony, if in addition it is planned to actively form various data entry forms using PHP - then Laravel. For the formation of a convenient and efficient interface on the client side today there are a large number of different frameworks. Among the most widely used can be called Angular and React.

As already noted, one of the basic requirements for all the tools used in the development of an information system for a university should be simplicity and efficiency, so that a small team of developers with minimal effort can create a high quality and convenient product in a short time. A distinctive feature of the university information system, as well as most corporate systems, is intensive work with a database using tables or hierarchical structures. The number of entries may be hundreds of thousands or more, which means that these components should support buffering, infinite scrolling, filtering and sorting, both client-side and server-side if necessary. It is also important that there are a large number of various components that provide convenient data entry, both in the form of forms and inside tables, for various types of data and with validation support. All these components must support integration and interaction among themselves "out of the box." In particular, they should initially be in the same style. Although these requirements may seem insignificant, in practice the integration of disparate components, both according to data, interaction, and by styles, can be a nontrivial task, which takes more effort than doing the direct work of creating a functional module or a specific form / table.

Unfortunately, most modern frameworks concentrate on the core and simple basic components, trying to make the most of the capabilities of modern HTML, which was not originally designed to work with "big data". There are a large number of additional components that implement complex work with tables, trees, etc., each of which is designed by different developers for different tasks. In practice, their integration can be quite difficult and expensive. Moreover, the use of various third party libraries, extending the basic capabilities of the framework, can lead to the application expanding to significant sizes, resulting in a dramatically increased load time and requirements for the client’s computer.

Therefore, to create a university information system, we chose the ExtJs javascript-based framework for developing corporate systems. This framework is commercial, but nevertheless, free versions under the GPL license are regularly provided, the community of developers exceeds 2 million. The main advantages of this framework are completeness and integration with support for high level components for working with "big data". In addition, the framework includes tools for developing and compiling production versions. The choice of this framework significantly reduces the need for finding and studying additional components and tools for assembling and compiling - they already exist in this framework and are integrated with each other. Dozens of different themes are also supported out of the box, including high contrast for people with low vision, night and others. At the same time, it’s not even the number of themes that is important, but the fact that all components support a single style system, so by changing the basic styles centrally, we can easily adapt the appearance of our application to the customer’s requirements without spending time tweaking the styles of the individual components. The framework continues to actively develop and supports almost all versions of modern browsers, including mobile ones.

Last year, the ExtJs Community Edition (CE) version was released, which is updated as often as commercial and is available free of charge if the profit from its use per year does not exceed $10,000, which is more than suitable for systems developed by universities for internal use. ExtJs has been developing for over 12 years and provides support for all modern browsers. Before the advent of full HTML5 support in all browsers, many complex effects could not be realized only by styles, especially in all browsers, so developers had to use more sophisticated methods using JavaScript. To maintain compatibility, components designed for browsers without HTML5 support are moved in the Classic Toolkit (not included in CE), and completely redesigned and optimized for HTML5 in the Modern Toolkit. Both toolkits support responsive configurations, allowing to create one application that adapts both to desktops, tablets and smartphones (among others, font sizes, indents, component appearance are automatically adapted), which allows you to create modern applications with support for all types of devices with minimal costs.

When developing a project that is constantly evolving and growing, over time, developers are faced with the fact that the size of the application becomes so large that the load time becomes unacceptable, and in some cases even lacks the resources to perform certain operations [2]. Starting with ExtJs 6.5, support for downloadable packages has been added both in the framework itself and in the Sencha Cmd command line tool. Although the CE tools do not have commands for working with packages, nevertheless the projects created with by them are fully compatible with Sencha Cmd (which is distributed free of charge), therefore the indicated possibility of compiling a project with packages intended for dynamic loading fully works for CE projects [3]. Thus, we can recommend create new projects based on the Modern Toolkit using Community Edition, and use Sencha Cmd for development (watch/build) and create packages. The application itself is only a loader + base packages used throughout the application, and all application functionality should be placed in dynamically loadable packages.

The central component of any information system is the database. The right choice of the RDBMS can determine the success of the entire project. Nevertheless, over the past 10 years, most popular RDBMSs have added support for virtually all the functions necessary to create a high performance corporate information system (transactions, support for referential integrity, stored procedures, including Java, high performance on complex multilevel queries from many tables and etc.). All RDBMSs already have advanced administrative tools with support for all major platforms. Among Open Source database initially most developed database was PostgreSql, but the lack of easy to use tools for Windows for a long time made it "exclusive" product for "professionals". FirebirdSql (originally Interbase) although it did not have as PostgreSql, object capabilities and an expanded set of built-in functions, initially supported all major relational database management system functions, and has a highly developed administrative tool. However, initially it was a commercial RDBMS, which limited the number of its users. MySql turned out to be the most popular and well known, which for a long time did not support high-level RDBMS functions, had poor performance on complex queries, but was cross-platform, had a large set of convenient tools and impressive performance when performing simple queries on simple unrelated data structures.

To date, the main functionality, different tools and performance of all these databases are generally comparable. Based on PostgreSql and MySql, you can build scalable load-balanced cluster. FirebirdSql does not support the creation of a load-balanced cluster (planned for the next release). Based on this, PostgreSql may be the best option for launching a new project, but if you have MySql or FirebirdSql specialists, you can safely choose these RDBMS.

Despite the fact that information systems should reduce the volume of printed documents, nevertheless, such documents are still not completely excluded. There are a number of forms of documents required for external reporting or requiring a signature to confirm authenticity. In this case, the information system can reduce the number of print versions by prior agreement and approval in electronic form, printing only the final version. When using the web interface, the task of the organization of printing documents arises. In the Electronic University, an approach that has already become widespread is used for this, when a document for printing is formed as a file on the server and sent to the client, where it is printed using locally installed programs. In order to simplify this process, most documents are generated in PDF format, since it is supported on almost every computer, and the format itself provides the same print result in different environments.

Although it is possible to simply create a PDF file using PHP, we consider it more appropriate to use special tools for design and report generation. One of the most suitable for this is JasperReports, which is a set of Java libraries for generating reports based on templates in XML format. One of the factors determining the choice of JasperReports is the availability of advanced report designer JasperSoft Studio. Since both the generator and the designer are written in Java, they are completely platform independent. We can generate and develop reports under Windows, and generate them on a Linux server. JasperReports comes with a large number of sample applications and reports. Thus, it is sufficient simply to create an application on their basis, which will integrate the report generator with the information system. Although JasperReports is not the only report generator on the market, and not unique to Java, its advantage over other generators, such as BIRT, is its extensive support for print reporting features (pagination). In particular, support for headers and footers, both for pages, groups and columns, various options for placing group totals depending on the totals of other groups and the distance to the bottom of the page and much more. The big advantage of using Java at the heart of the report generator is the possibility of using Java expressions inside the report and the introduction of custom Java classes, which opens up virtually unlimited possibilities for generating a report even in the most difficult cases. It also supports nested reports, cross tables, subtables and diagrams with the ability to transfer parameters to subordinate data sources (subreports, crosstables, etc.) and return the calculated values to the main report. As a data source, you can use as any DBMS via JDBC, as well as text files, JSON, XML, XLS, XLSX, ODS. It is also possible to create your own data providers to access any data source. Although the main interest for us is the export of the report in PDF format, the formats DOC, DOCX, XLS, XLSX, ODS, ODT, HTML are also supported. Thus, it is possible to form a fairly complex report, available for further editing / filling.

## III. Experience in Designing an Information System

We started the project of our information system Electronic University, based on the following provisions. It is necessary to quickly create most basic modules that would allow the users of our university to begin work on filling the system with data. In the future, additional modules are added to the system, expanding its capabilities in accordance with accepted priorities. The information in the database should reflect not only the current state of all information objects, but also the complete history of their changes. In this case, with the help of certain queries, we can get / calculate the state of any object at almost any point in time, which is very important for the work of our university [4].

Traditionally, databases are divided into operational and analytical. Operational databases process data in real time and store only the current state. Analytical databases stores and quickly processes historical information, and receives information not in real time, but as a result of periodic synchronization with operational databases, receiving, as a rule, already aggregated information. It’s often limited to a specific (reporting) period, for example: day, month, quarter. The proposed approach allows storing historical information in an operational database with virtually no loss of performance, receiving the current state in real time based on historical data, and performing analytical processing on their basis. As a result, there is no need to maintain two different databases. Moreover, analytical processing can be performed at any time based on current, relevant data, without the need for additional synchronization.

The basis of the information system is the activity of people in the educational process [5], [6]. Therefore, the first basic modules of our system were the “HR department” and “educational department”.

The basis of the “HR department” is personal data about people (employees and students) and administrative orders reflecting documents for any moment of changes in the position of people in the service or study structure. We have developed a whole system of orders for the reception, dismissal and relocation of employees. A similar system of orders was developed for enrollment, deduction and transfer of students to the appropriate course, specialty financial form, etc [7]. Each electronic order is an entry in the corresponding database table. Any new state of an employee or student is also an entry in the subtable of the corresponding order. The duty of the HR department staff is in time create electronic orders for any changes in the personnel structure of staff and students. Any other module of the system that operates with lists of employees or students, refers to the HR department module and requests the appropriate list for the required date. The system calculates the status of each person from the list on the basis of electronic orders. For example, if an order is introduced to dismiss some teacher or dismiss a student from tomorrow, then today this person will still be in the list of the relevant department or study group. Tomorrow, when calculating the list of this department or study group, the system will automatically exclude such a person from the list. At the same time, for any employee or student, you can get complete data along with the history of his personnel changes. Similarly, you can get a list of employees or students for any past or future date. For example, you can easily get a list of a group of students who graduated last year. This approach very easily allowed us, for example, to form a separate module for all university graduates.

The activity of the education department is based on the curriculum. We have developed an electronic curriculum structure that matches the patterns approved by the ministry and the management of our university. The curriculum for each individual specialty is entered into the database. Changes and correction of plans for each academic year are recorded in the form of so-called work plans. On the basis of work plans, individual curricula for each student are formed annually in the dean's offices. On the basis of the students' individual plans, an electronic lesson schedule and session control for each academic discipline is formed [8].

The next most important module that could already use data from personnel and the training department is the dean's module. The structure of the dean's office was made up of electronic sheets for session control (exams, tests, course projects) and students' individual plans. The dean's office receives lists of student groups from the personnel department module. They are simply calculated on the basis of orders for the date requested by the dean's office. All sheets are formed and associated with the individual plans of students. After making session assessments, they are automatically displayed in the individual plans of each particular student. For deans, a general table of the results of any session can be formed for groups or specialties of students. In our system, electronic statements record any fact of passing or re-passing any type of control over academic disciplines. Therefore, the system can calculate the status of the results of the session for any student at any time. This allows deans to see and analyze the dynamics of the results during any session. A number of additional functions were added to the dean's module in the process of further development of the system (for example, lists of debtors based on the results of sessions, averaged rates of passing sessions by groups or specialties, generation of data for printing applications to diplomas, etc.).

The next module is the “Chair”. It allowed the responsible staff of the department on the basis of the curriculum to carry out the assignment of disciplines to teachers, as well as monitor the results of the sessions in all disciplines of the department. Based on this information, the function of creating an electronic lesson schedule has been added to the module of the training department.

Next, we have developed modules "Teacher" and "Student." For teachers, we have connected the possibility of individual filling out electronic sheets for the assigned disciplines formed by the deans. Also, teachers were able to view the class schedule, the list of disciplines of the department for any semester, which was formed by scanning the curricula of all specialties on the basis of the assigned department. In the future, the teacher module was significantly expanded by the subsystem of the rating system of teachers, developed by the staff of our university. Further, an application developed on the basis of the ExtJs framework was added to the module, which allows automating the accumulation of data from the daily activities of each teacher - an “electronic journal”. The teacher periodically assigns marks for all types of students' activities. The total grade for the discipline for each student is automatically calculated in an electronic journal based on the formulas developed by the methodological management of the university, according to the weights of each type of activities. Also, the ability to form a work program for each discipline assigned to the teacher has been added to the teacher’s module. All information from electronic journals or work programs is available to deans and students.

The student module allows each student logged into the system to view their personal data, their individual curriculum, grades in the electronic journal for all classes, results of sessions, a schedule of classes. In the future, the function of forming an individual curriculum by the student himself was added by selecting certain disciplines from the respective lists offered by the chairs and deans.

In the future, the system can be expanded with additional modules, thus integrating the activities of new university services into an electronic information system. To do this, add the corresponding structure to the database (usually one or several related tables) and the corresponding module to form the server and client interface. Usually, the relevant responsive staff of the university turns to us, which, for example, want to see and analyze the relevant activities in the educational process. Naturally, the basis of such information are the structures developed for the modules of the HR department, educational department, dean's office and teachers [9].

## IV. Cluster Creation Experience

After the introduction of the university management information system and the completion of the user adaptation period, a very crucial issue is the reliability of the system and the high accessibility of users to the system 24/7. We will assume that the network functions reliably and its throughput ensures all the needs of the system. Therefore, we will discuss the main aspects of the software and hardware architecture of such a system [10].

Experience in operating an information system at our university showed that when placing the main components of an information system (web server + database server) on a single server, there are the following problems that do not allow for constant round-the-clock access to the system.

Updates of the operating system and particular services. In this case, the server has to reboot or temporarily suspend the updated services of the information system. This is not a big problem if administrators perform updates at a time that is not critical to the work of users.

Any system sometimes fails, or after a software update, problems suddenly arise related to the modified parameters of the updated components. In these cases, the system may be temporarily unavailable, sometimes (depending on the complexity of the problem) even for a very long period.

After aging hardware component dramatically increases the probability of failure in the system. This is especially true for hard drives. In this case, any operating system can begin to behave quite inadequately and at the same time to discover the real cause of the problem that has arisen can be quite difficult. Once we had damage on our hard drives. As a result, the system continued to work, but the speed of work fell sharply, the CPU load for no apparent reason increased to almost 100%. Identify the problem was not easy. The whole working day of administrators and users of information system has turned into torture.

For reliable operation of the equipment, it is necessary to periodically conduct preventive maintenance work. For a single server, this also means an interruption in the work of the information system.

And, finally, an elementary excess of server load, when a single server simply cannot cope with a large number of users accessing it simultaneously or in a very short period of time.

The basic recipe for improving the reliability of systems in such situations is duplication and redundancy of both hardware and software components. Of course, for reliable operation of hard drives, it is necessary in such cases to use RAID arrays of at least level 1 (mirroring). In case of software failures, the damaged disk automatically leaves the array and the system sends the corresponding message to the administrator. In this case, you can safely perform the restoration work and reinsert the disk into the RAID. If the disk completely fails, the system may also have a small interruption (shutting down the server and replacing the damaged disk with a working one and then inserting it into the RAID array).

To duplicate the remaining software and hardware components of the system, today we can find quite inexpensive cluster solutions. Cluster organization requires at least two hardware servers and corresponding cluster management software. At the same time, two or more physical servers are visible to users as a single system in terms of requests and obtaining relevant information.

The simplest cluster is a “high availability” cluster, when all the main services of both servers duplicate each other, but only one is active. Second server is constantly running and is in hot standby. Cluster manager monitors all services of the active server. In case of stopping or out of any service, the manager automatically switches this service to the backup server and the system continues to work. In this case, it is very convenient to do software updates or maintenance work. To do this, it is enough to remove a separate server from the cluster, perform all the necessary work on it and reenter it into the cluster. Then a similar procedure can be performed with another server in the cluster.

However, for high availability (HA) clusters, the problem of redistributing the load if it is exceeded resources of the single server is not solved. In such cases, a so-called load balancer is added to the cluster. The classic cluster diagram with a load balancer is shown in the following figure (Fig. 1).

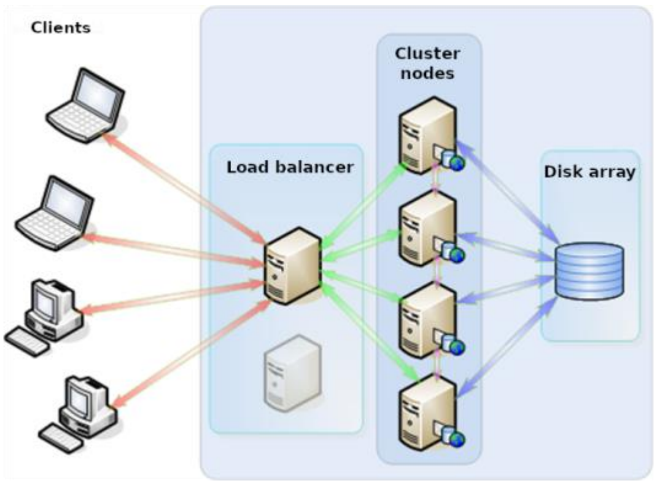

Fig. 1. Classic cluster with load balancer

Here, the load balancer is a separate server that users see as the “entry point” to the system. Cluster nodes are separate servers that duplicate various components of the system. A very important point is the “disk array” that all nodes of the cluster must “see”. At the same time, each individual node gets access to the same system information on disk (for example, a site or database). To improve the reliability of the system, the load balancer must also be duplicated, that is, a separate failover cluster can be organized specifically for the balancer.

Such a solution can be quite expensive, especially if you have a separate hardware disk array. Can we just get by with two separate servers and get a failover cluster with load balancing? It turns out you can.

For our system, if there are two identical servers (by the way, in general, servers may differ from each other in hardware configuration) we used the Linux operating system (CentOS-6.5) and the cluster management software offered by clusterlabs.com. This solution is open, free and fairly reliable.

The pacemaker cluster manager is installed on both cluster nodes, which uses the corosync service to monitor individual services of the system. Each cluster node is set to a static real IP address. You also need a third IP address that users will contact and which will be “floating” between the individual nodes. The pacemaker manager provides a special IPAddr resource that is installed on both nodes, but is activated only on one of them. We also assign the third “cluster” IP address to this resource. Also, load balancers are installed on both nodes (for example, HAProxy for a web server or PgPool-II for a PostgreSQL database server). The balancer is associated with a cluster IP address. Therefore, only one balancer will also respond, the second will be in reserve. In the configuration of each balancer, it is indicated that the load is divided between the corresponding services of both nodes. Web services and database services are installed on both sites with the same configuration. At the same time, the pacemaker ensures that when the cluster is raised, these services are activated on both nodes.

Further work scheme is quite simple. Let, for example, the cluster IP address (and therefore the load balancer) are up on the first node. Then he will answer all users. Upon receiving a request, the balancer redirects it either to the service of the first node (to itself), or to the service of the second node (to static IP addresses), ensuring that the number of requests is distributed evenly. Since the main functions of the balancer (frontend) are tracking and redirection, a separate balancing service is able to handle a huge number of requests from users. The main load falls on the end services (backend), between which requests are redistributed, which means that the load capacity of the cluster increases at least 2 times. If, for example, one of the cluster nodes needs to be sent for maintenance, it's simply removed from the cluster. At the same time, all services automatically migrate to another node (along with the cluster IP address and balancer). Now users will only answer the second server. The balancer will determine that only services on itself are "alive" and all requests will be redirected only to them. In fact, now the cluster works as a single server. But users will notice almost nothing, especially if the load on the cluster at this moment is small. We quietly deal with the first server and after all the work is finished, we load and enter it into the cluster. The balancer, having discovered the presence of additional services, starts regular load balancing.

In the same way, another cluster node can be set up. At the same time, the entire system remains permanently accessible to users. As you can see, the combination of redundancy and load balancing allows you to significantly increase both the performance of the entire information system and increase its reliability by providing constant round-the-clock user access with only two separate servers.

The most "fragile" cluster element in our case is the disk array. We do not have a separate hardware device. Therefore, a special cluster wide RAID 1 level is organized using the DRBD service, in which each node has its own mirrored disk, which is automatically replicated at high speed via additional gigabit network interfaces with any changes on any of the disks. Thus, each node has its own copy of the shared disk storage, which is almost instantly synchronized with the other node. It is "almost" and is thus a fragile element.

## V. Conclusions

The described cluster solution has been successfully used by us for the past 6 years and has shown very high efficiency and reliability. The inaccessibility of the information system for users mainly arises due to problems in the internal network or the network of our provider. Proposed approach to database structure allows combine analytical and operational data processing in one database, getting current state from historical data in real time and allowing analytical processing at any moment on current relevant data without any delays for synchronization. Continuous development over time leads to big application footprint, but proposed set of tools and dynamic package loading based on CE combined with traditional Sencha Cmd builds solve this issue. Instead of proposing PHP as main development language for web development, we recommend Java based Jasper Report tools and libraries for report generation. Integration of PHP with other tools on back end allow significantly improve overall application performance. Instead of using expensive hardware cluster solutions proposed software high availability cluster with load balancer configuration, that allows to build clusters on existing hardware with at least two nodes without additional expenses.

Also this allows gradually scale up the system adding new nodes without need of full hardware replacement at once. Thus, 20 years of practical experience in developing an information system for a university has shown the viability of the concepts and principles of designing modules of the system we have adopted. Today, most of the processes of activity at our university are integrated into the Electronic University. We can use new modern development tools and introduce them into our system, adding new modules or gradually updating existing ones. Our approaches will allow the following developers, who will replace us in the future, to continue the development of the system, gradually expanding it and modernizing it.

## References

[1] P. Nicolaescu, M. Rosenstengel, M. Derntl, R. Klamma, and M. Jarke, "Near real-time collaborative modeling for view-based Web information systems engineering", *Information Systems*, vol. 74, p. 1, pp. 23-39, May 2018

[2] A.S. Lee, M. A. Thomas and R. L. Baskerville, "Going back to basics in design science: from the information technology artifact to the information systems artifact", *Information Systems Journal*, vol. 25, iss. 1, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/isj.12054. Accessed on: July 10, 2019

[3] Xuequn Wang, Xiaolin Lin & Nick Hajli (2019) Understanding Software Engineers' Skill Development in Software Development, Journal of Computer Information Systems. Doi: 10.1080/08874417.2019.1566805

[4] A.Y. Mazarchuk, C.E. Belovsky, and S.S. Grygoruk, "Structure, architecture and functionality of modern integrated university management system", in *Actual problems of traing specialists in ICT. Conference proceedings*. - Sumy: Sumy State University, 2013, p. 2, pp. 22-29

[5] Albert H. Huang, "A Model for Environmentally Sustainable Information Systems Development", *Journal of Computer Information Systems*, vol. 49, no. 4, pp. 114-121, 2009

[6] Tim Huygh and Steven De Haes "Investigating IT Governance through the Viable System Model", *Information Systems Management*, vol. 36, no. 2, pp. 168-192, 2019.

[7] S. Henningsson, P.W. Yetton, and P.J. Wynne, "A review of information system integration in mergers and acquisitions", *Journal of Information Technology*, , vol. 33, no. 4, pp 255–303, 2018.

[8] R. Hadidi, and D. Power, "Management Information Systems (MIS) Curricula Development, Management, and Delivery - Possible Sharing Economy Solutions," *Journal of the Midwest Association for Information Systems*, vol. 2019, iss. 1, 2019. [Online]. Available: https://aisel.aisnet.org/jmwais/vol2019/iss1/1. Accessed on: July 19, 2019.

[9] C. Bel'ovsky "Designing a Modern Information System of University Management: the Experience of Khmelnitsky National University", *Journal of Research on Trade, Management and Economic Development*, vol.2 , iss. 1, pp. 63-69, 2015.

[10] A.Castellanos, J.Cigarran, and A.García-Serrano, "Formal concept analysis for topic detection: A clustering quality experimental analysis", *Information Systems*, vol. 66, pp. 24-42, June 2017.