Introduction

Usage of information technologies leads to employment of technologies in every branch of life, while all these implementation causes to rapid development of human life. Results of such development we can notice paying attention to the rate of used technologies for daily demand in developed countries, of course such progress all members of society has to be ready to changes both economically and traditionally.

One of the five main concepts of Uzbekistan's market economy – dominance of government in each novelty, thus each strategic plan will be build and supported by government, taking into account status of society, traditions, geographic position, citizen demand and all other criteria. Now one of the main points of development is Information Technology (IT), whilst XXI century is "Information Technologies century" and paid attention to this field is greater in our country too. Confirmation of sentence can be done paying attention to recently accepted government laws, while a number of them were accepted, especially considering to IT, such as "On electronic commerce" and "On electronic document flow" law of Uzbekistan /1/, "On digital signature" /2/ and "Organization cryptographic protection of information in Uzbekistan Republic" /3/.

If we carefully look to the acceptance order of Information Communication Technologies (ICT) laws, we can notice, that all laws serves to rapid development of e-commerce, while most of laws considering production serves as a ground for e-commerce applications. These cases show great importance of this field for government.

As society moves to paperless document, role of e-document exchange increases. Because all business and government offices will exchange with e-document, people's adaptation to e-documents and to e-displays will rise, where all of these leads to paperless office and of course paperless office is directed to e-Government, in this final qualifying work overall status of e-Government projects are discussed.

The **importance of final qualifying work** is development of secure e-document exchange, while as was defined in “about e-commerce” law of Uzbekistan “e-commerce is – any form of business transaction in which the parties interact electronically rather than by physical exchanges or direct physical contact” /1/. In real business process or even in simple information exchange both sender and receiver exchanges piece of information, where more complex business procedures may require electronic document exchange in this purpose. However, none of business processes can be done without message or document structure agreement. Final qualifying work provides solution to common e-document exchange, additional to provide reliability and completeness of system a number of cryptographic tools such encryption, hash function, digital signature and data compression tools are used, whilst all of these features offer better opportunities to information society.

Fully validity and juridical force of e-document as in hard paper, can be gained with usage of cryptographic tools, whilst additional to provide secure e-document exchange system, this final qualifying work discusses worldwide requirements and recommendation to e-document formats, fine grained e-document exchange flow also.

Final qualifying work consists of five chapters. First chapter is about overall status and analysis of e-document exchange, where requirements, advantages and obstacles, international standards, formats, recommendation for of e-document exchange, and e-Government projects of different countries and role of e-document exchange systems in it are explored, finally defining purpose, scope and expected results of final qualifying work. Second chapter is devoted to structure of e-document exchange system, whilst architecture, applied cryptographic tools and new cryptographic algorithm on working with confidential information in e-document exchange system. Third chapter is about creating web solution, applied cryptographic tools on working with confidential information in e-document exchange system, where used technologies, working environment, developed cryptographic libraries, document flow, complexity and robustness of different

crypto algorithms are given. Fourth chapter is devoted to exact recommendations for employing objects and usage sphere of solution. Finally last fifth chapter is about safety precautions.

Taking into account all above discussed options and learning existing international solution, on the scope of this final qualifying work we propose web solution, where main cryptographic tools such encryption, hash function, digital sign and data compress are employed to provide confidential e-document exchange. Further development trends and real deployment objects are proposed at the content of final qualifying work.

I. Overall status and analysis of electronic document exchange

XX century – as an *industry* century bring human being to new stage of life, where human invent *electron* and use, manipulate this small bit in his own interest. Next simple electronic devises such as radio, TV, phone and etc. was discovered and latest essential invention was computer, while this thing was expected to be in wide use. After some time, computers replaced most of electronic devices, including radio, TV and even phone, being not a just local conversation tool, but global information exchange device and multi-function multimedia device.

Computers leads us to better, stable, dynamic and reliable lifestyle, while when typing terminals in governments agencies and private offices were replaced by simple keyboard enabled computers, there were dramatically income in stuff, time and paper, and of course typists were able to easily correct there letters or update content of their message, instead of wholly type document again.

Nowadays, computer world come to new era, called network based computers, while internal and external global networks, such as *internet* gives us unbelievable opportunities.

This chapter devoted to look to history and analyze current stage of computer technologies, worldwide web, e-commerce and especially electronic document exchange systems.

In Chapter 1.1 we'll learn about electronic document itself and real requirements for electronic document exchange system. Giving national and international definitions to *electronic document*, its content, validity as a paper signed documents and overall information about obstacles of wholly implementation of electronic documents.

Chapter 1.2 gives information about requirements to the structure and content of electronic documents and what kind of recommendations, proposals were constructed recommended to use in European Union, United States and worldwide.

Chapter 1.3 devoted to discussion and comparison of the widest implementation of electronic document exchange system, *e-Government* project of countries. While this project is every country's pointing project and there are a number of prerequisites for its realization, we'll look just in 7 different criteria's and 2008 April report of worldwide e-Government project experts.

Analyzing and concluding previous chapters, next Chapter 1.4 defines purpose, scope and expected result of this final qualifying work.

1.1. Requirements for electronic document exchange, advantages and obstacles

Apart from vastly improving the efficiency of document processing, the application of information and communications technology, the adoption of innovative administrative procedures, the computerization of document production and management, the automation of investigation and tracking tasks, and the use of the Internet to exchange official documents among different agencies are all helping to further high-quality, high-efficiency e-government services. Here the transmission of documents between agencies via electronic document exchange has delivered some of the most obvious benefits: While it once took at least one or two days to transmit an official document, now it takes only 10 or 20 seconds. This great breakthrough has thus increased efficiency a thousand-fold.

Electronic document exchange has been a great step forward in the modernization of official document processing. Beyond heralding the emergence of computers and the Internet, this advance also signifies that the pens, paper, and other office tools long used by civil servants have been replaced by the computer mouse, screen, printer, modem — the new "four treasures of the scribe." This is a sign of the sweeping changes that have been made in government operating procedures. By laying a sturdy foundation for e-government, electronic document exchange has brought the new era of digital administration that much closer.

The Importance of Interoperability. Web services integration solutions aggregate trading partners through a single point of connection. Utilizing a hub-and-spoke approach, these hubs perform all necessary data mapping, security, contractual performance assurances and maintenance to allow customers and partners to communicate with any number of other customers and partners. The cost savings attributable to the hub-and-spoke approach are significant, and grow exponentially as the number of business partner connections increase /26/.

Companies must continue to push towards interoperability in order to drive repetitive costs out of their organizations and gain efficiencies. Technologies have advanced to the point where the pieces to conduct e-commerce are available. The challenge is aggregating them in a flexible, scalable and manageable way that is cost-effective for independent corporations. It is important to understand what these Web service integration platforms provide for corporations as follows.

- Ability to transact electronically with any trading partner at any time – companies cannot afford to limit their trading partner relationships to those with which integration is easy. Business document exchanges allow organizations to conduct business with any partner in any geographic region. For example, a small business may connect to the electronic document exchange to trade with its major supplier. By virtue of a single connection, the small business would be connected automatically to banks, payroll providers, shipping companies and trading partners. Previously, small businesses would never have been able to afford a direct electronic connection. For a low up-front charge and a monthly subscription, the electronic document exchange manages the connection to all trading partners. Risks are low as companies have the ability to quickly connect and disconnect with any trading partner.

- Focus on growing revenue and improving supply chain efficiencies – a single connection to a business document exchange will allow integration to other trading partners that are already connected. This will allow companies to provide products and services to a wider range of trading partners quickly and easily. Companies achieve significant cost savings and efficiencies because

they are able to reduce order and sales cycle times, keeping inventory levels low and reducing errors due to duplicate keying of information.

- Increased return on technology investments and rapid implementation – Web services integration platforms support multiple connection protocols and data format types. This will allow companies to maximize the returns on existing investments by extending the benefits provided by those applications to a company's customers, suppliers and other business partners. These managed service solutions isolate companies from the risk associated with changing standards and technology used internally or by their trading partners.

- Drive down operating and maintenance costs – a managed service solution for the exchange of electronic documents provides a number of key benefits. Companies do not have to incur expensive and complex implementations of technology. Using a managed service significantly drives down the average cost of labor, implementation, technology and operating costs that are incurred typically to implement and maintain direct point-to-point trading partner relationships.

- Access to information beyond the supply chain – information is consistently sent to organizations that cannot be utilized because the data is not in a format that internal systems can understand. Electronic document exchanges can transform this information into a usable format that can then be read and integrated with internal systems.

Given the advantages of a paperless office, why is society not yet there? The answer is deceptively simple: There is no substitute on the market today that is as portable, durable, simple, and accessible as a paper. Canada is the world's largest exporter of office printing paper, and they have seen their paper exports more than double in the last 15 years, the same time frame as the computer revolution. There seems to be no retarding the propagation of paper in the office, at least not in the near term. In fact, Hewlett Packard, one of the world's most well known printer manufacturers, predicts a 50% increase in paper use over the next

five years. One of the primary contributors to the rise in paper consumption is the propagation of computers and computerized data communication.

Over 200 million printers have been sold since 1998 and some 6.3 million printers designed to print digital photographs will be sold by 2003. Additionally, Gartner Group has estimated that laser printer and fax machine sales in the United States have increased 12 and 22 times over, respectively, during the 1990s. The rise in printer sales have to do primarily with the converse drops in cost for these peripherals. Each of these devices requires paper, and for 2001 it is estimated North America will use some 2.2 trillion pieces of paper in printers, fax machines, copiers and other document-reliant machines.

Printed words on paper seem to add some credibility and permanence to the information being transpired. Computers and high capacity printers "have increased paper usage ten-fold," according to Foote, "simply through their ability to spew out copies faster and more furiously than ever". "Just about every innovation in the digital revolution was supposed to cut out more paper," reads the year 2000 annual review of the Forest Products Association of Canada. "Precisely the opposite continues to happen". Paper is a medium which everyone can use and to which everyone has access.

Electronic Document Exchange Services. Today, e-business technology and standards are in a constant state of change. The market is consolidating quickly and larger organizations are moving to put their mark on technologies that facilitate business-to-business integration (B2Bi). These organizations are leading an effort to drive the development and acceptance of Web application servers as development platforms for the next generation of conducting e-commerce. As companies try to understand and determine the competitive strengths and weaknesses of these solutions, they may be losing their competitive edge.

Management and integration of information between business partners continues to be a significant challenge across multiple businesses and industries. Despite advancements in technology and the emergence of standards, companies

are faced with difficult decisions on how they can integrate their system internally and externally to gain cost and competitive advantages.

An electronic document exchange, otherwise known as a Web services integration solution, will allow organizations to integrate with their trading partners quickly and cost-effectively. Business documents encompass a large set of transactions that can be automated and include purchase orders, sales orders and inventory items, but can also include other important data or information that needs to be transacted between business partners.

Key Business Benefits. The use of an electronic document exchange enables the secure movement of data between independent organizations and provides a quick and easy way for companies to move towards e-commerce. Organizations providing these services focus on the scalability of costs and technology and leverage the Internet to facilitate the flow of information between organizations. These centralized electronic document exchange companies base their technology on developed and emerging transactional standards and Web application server platforms. Businesses that access these platforms do not have to worry about disparate data standards, security solutions and technology to transact electronically with their trading partners. The electronic document exchange company acts as a managed service to isolate customers from the risks associated with managing the technology and the emerging standards that are used among multiple trading partners.

An electronic document exchange provides companies with the ability to:

- connect once to a central hub allowing quick and easy integration to multiple trading partners;
- focus on building their business by connecting quickly and cost-effectively with their trading partners;
- extend the return on investment of their existing systems;
- drive down the operating and maintenance costs of multiple trading partner integration;

- utilize data from external sources that is transformed into formats that can be understood by internal applications.

Third-Party Central Processor. Electronic document exchange may involve a third-party central processor, which is to say that electronic document exchange procedures may be carried out by an electronic document service center. In contrast, direct point-to-point transfer means that the sender and recipient use the Internet to directly transmit documents without the involvement of any third parties. It is essential that these two means of transmitting official documents offer reliability and security. Apart from such basic requirements as affixing and confirming signatures, separate transmission of original documents and copies, and automatic response on receipt by the recipient, individual agencies may require additional value-added services reflecting their particular needs or document types among electronic document exchange functions /27/.

1.2. Analysis of international standards, formats and recommendation for electronic documents

To integrate and associate different formats of e-documents active web controlling organizations proposed general specifications of e-document, where all of these specification gives general requirements, which e-document has to support to maintain easy storage, manipulation, retrieval.

Here we'll describe three worldwide used formats of e-documents, while first is Java Specification Request 170 (JSR 170) /21/, which was proposed by Java Community Process (JCR), Electronic Records Management Software Applications Design Criteria Standard (DoD 5015.02-STD) which were recommended from United States agency defining general model for e-documents April 2007 year, and Model Requirements For The Management of Electronic Records (MoReq Specification) which has been prepared for the Interchange of Data between Administrations (IDA Programme) of the European Commission by

Cornwell Management Consultants plc. The format specification is designed for simplicity and ease of implementation at the lowest possible cost /21/.

JSR 170. The World Wide Web is the most pervasive software system ever developed. The Web uses a simple, standardized interface to encompass information from all over the world regardless of how that information is created, stored, and processed behind the interface /26/. As a result, Web-based services can be implemented on any form of computer system, whether or not they are connected to the Internet.

Application development doesn't have to mirror the complexity of its applications. The web has shown that complex goals can be achieved using a very simple interface based on content-centric design and a commitment to standardization. The same architectural principles that made the Web successful can be applied to application development within servers.

A content-centric interface eases the task of application integration by focusing on the uniform nature of content rather than the specific controls of any given application. To illustrate the difference between content-centric and control-centric interfaces, consider the task of integrating with a word processing application. A control-centric approach would be to look at the functionality provided by the command menus of the word processor, such as "Format/Paragraph…," and provide method interfaces that replicate the commands and data-entry dialogs of the word processor. A content-centric approach would be to focus on the data managed by the application: in this case, a sequence of paragraphs with associated formatting.

The control-centric approach is able to take advantage of the unique behavior and functionality built into the word processor, but that comes at great cost: the API will consist of hundreds of actions that are tied to a specific version of one vendor's word processor. In contrast, the data-centric approach limits functionality directly obtainable via the API, but enables an unlimited number of additional tools to be applied to the common data model, eventually surpassing the functionality that could be provided by any single vendor.

Following the Web's architectural principles in designing a content-centric interface does not imply we are limited to the protocols and data formats that make up the Web interface (e.g., HTTP and HTML) instead, we can simply learn from the design principles of the Web and its focus on uniform identifiers, standard methods, and extensible representation types /6/. Interactions between Web clients and servers consist of course-grained messages exchanged over high-latency networks. In contrast, application development within a server consists primarily of fine-grain interactions upon local data stores. Therefore, what is needed is a simplifying architecture that promotes content-centric design via a uniform interface, and yet one that is as suitable for tiny data interactions as it is for multi-gigabyte data transfers. We refer to that interface as a Content Repository API.

A content repository is a generic application data "super store." In addition to being adept at handling both small and large-scale data interactions, a content repository is expected to manipulate and store structured and unstructured content, binary and text formats, metadata, and relationships that vary dynamically. Supports for advanced content services are also desirable, such as uniform access control, locking, transactions, versioning, observation, and search.

Content Repository API for Java Technology (JCR). The Content Repository API for Java Technology (JCR) is an ongoing effort to define a standard repository interface for the J2SE/J2EE™ platforms /12/. The goal of a content repository API is to abstract the details of application data storage and retrieval such that many different applications can use the same interface, for multiple purposes, without significant performance degradation. Content services can then be layered on top of that abstraction to enable software reuse and reduce application development time.

A traditional application uses multiple data stores during its operation. For example, a typical email application will store its configuration in a property list, its address book in a table, messages within indexed files (folders), message properties in separate tables, and search indices in a binary hash. In most cases, each of those storage formats would have their own interface. The application

developer would spend a significant portion of the development effort designing, creating, and maintaining those interfaces.

In contrast, a content repository API separates the issues of content storage and efficient retrieval. The application developer defines how the content is identified via the interface, writes the content, and then uses the built-in services of the API to perform efficient retrieval in a variety of modes: individual reads, traversals of related data, hierarchical search, and full database query. The real storage format is separated from the application interactions, allowing the most appropriate storage subsystem to be selected based on observing the actual performance of the application, rather than by making a premature decision early in the application's design. The application developer doesn't have to worry about parsing file formats, maintaining search indices for text content, managing transactions, or exporting data between applications; content services like those can be provided by a repository API without being specific to the application.

Repository Model. The repository model consists of an unbounded set of named workspaces, with each workspace containing a virtual hierarchy of items in the form of a tree of nodes and properties. Nodes provide names and structure to the content while properties contain the content. The easiest way to visualize a JCR workspace is through comparison with the UNIX file system structure, which consists of a tree of directories and files. However, there are some distinct differences.

JSR 170 and the Java Community Process. Industry standards for the Java™ language and J2SE/J2EE™ platforms are created within the Java Community Process /10/. Members of the JCP participate in the proposal and development of community specifications, referred to as Java Specification Requests (JSRs).

Optional Functionality. JCR provides a standard interface to additional content services as optional functionality

- **Locking** allows a user to temporarily lock nodes in order to prevent other users from changing them.

- **Transactions** may be supported through adherence to the Java Transaction API (JTA) /27/. JTA provides for two general approaches to transactions: container-managed transactions and user-managed transactions. A JCR implementation must support both of these approaches if it provides the transactions feature.
- **Versioning** allows the state of a node to be recorded in such a way that it can later be viewed or restored. The JCR versioning system is modeled after the Workspace Versioning and Configuration Management (WVCM) API defined by JSR 147 /25/.
- **Observation** enables applications to register interest in events that describe changes to a workspace and then monitor and respond to those events. The observation mechanism dispatches events when a persistent change is made to the workspace.
- SQL search provides an additional query language beyond the XPath search /15/.

Optional features allow for a variety of repository implementations while retaining a single API for application development. Application deployment specialists can select repository capacity and feature sets based on the application's needs, rather than some pre-determined level of functionality, thereby reducing costs and minimizing complexity.

Electronic Records Management Software Applications Design Criteria Standard. DoD 5015.02-STD April 25, 2007. DoD 5015.02-STD standard is reissued under the authority of DoD Directive 5015.2, "Department of Defense Records Management Program," it sets forth mandatory baseline functional requirements for Records Management Application (RMA) software used by the DoD Components in implementing their records management programs; defines required system interfaces and search criteria that RMAs shall support; and describes the minimum records management requirements that must be met.

- Managing Records. RMAs shall manage records in accordance with this Standard, regardless of storage media or other characteristics Code of Federal Regulations section 1222.10.
- Accommodating Dates and Date Logic. RMAs shall correctly accommodate and process information that contains dates in current, previous, and future centuries. A recommended format is, e.g. YYYY-MM-DD. RMA shall provide date translation from other date formats.
- Meta -Tagging Organizational Data. RMAs shall allow for the implementation of discovery meta-tagging /15/. This requirement implies the capability for adding Organization-Defined metadata fields, modifying existing field labels, and mapping data fields to standard transfer format fields.
- Backward Compatibility. RMAs shall provide the capability to access information from their superseded repositories and databases. This capability shall support at least 1 previously verified version of backward compatibility.
- Accessibility. The available documentation for RMAs shall include product information that describes features.
- Extensibility. RMAs shall include the capability to provide open standards interfaces in order to integrate into an organization's information technology enterprise. This capability shall include the capability to accept and file records from producing applications and provide support to the organization's workflow.
- Security Compliance. RMAs shall support applicable security standards including Security Technical Implementation Guides.

Here we show only general requirements, while each of these criteria define proper explanation of implementation and usage, referring to different Federal Regulations sections, while this specification express all features which Record Management Application had to support and general requirements for e-document formats.

Model Requirements for the Management of Electronic Records (MoReq). This specification has been prepared for the Interchange of Data

between Administrations (IDA Programme) of the European Commission by Cornwell Management Consultants plc. It focuses mainly on the functional requirements for the management of electronic records by an Electronic Records Management System (ERMS) /31/.

This specification is written to be equally applicable to public and private sector organizations which wish to introduce ERMS, or which wish to assess the ERMS capability they currently have in place.

The requirements in this specification are intended to serve as a model. They are not prescriptive for all possible ERMS implementations; some requirements will not apply in some environments. Different business sectors, different scales, different organization types and other factors will also introduce additional specific requirements. This specification must therefore be customized before use.

Each of specification consists of more than hundred pages content, while full presentation and discussion of content unfeasible in this Chapter, we will explain general concepts Figure 1.1 and describe only some characteristic which has to be provided by e-document format Figure 1.2.

For instance specification say "…This specification makes no assumption about the nature of individual Electronic Record Management System (ERMS) solutions. Users of this specification will need to determine how the functionality of an ERMS can be implemented to meet their requirements."

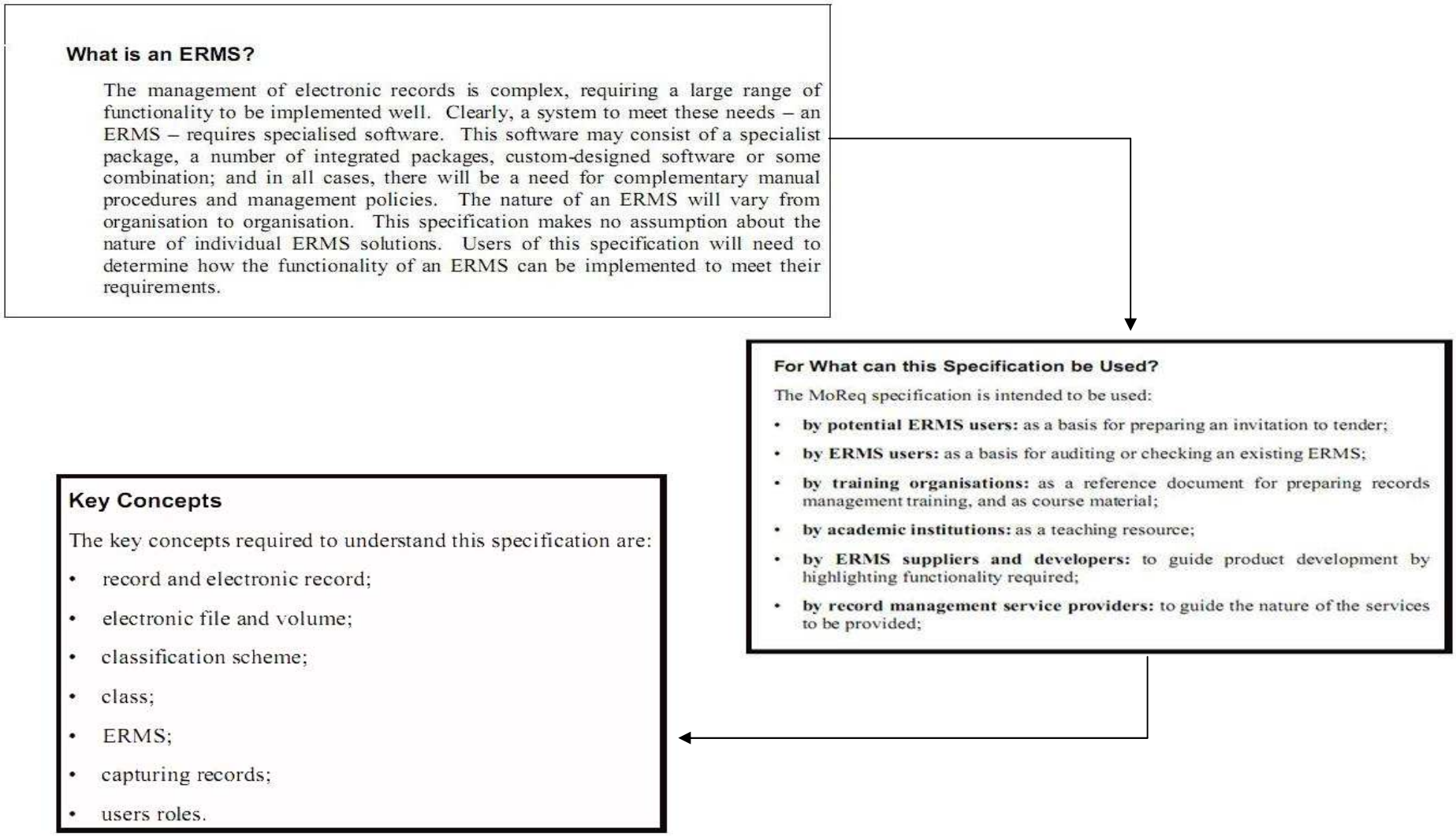


Figure 1.1. Terms, general purpose and key concept of MoReq specification

3.1 Configuring the Classification Scheme

- 3.1.1 The ERMS must support and be compatible with the organization's classification scheme.
- 3.1.2 The ERMS must be able to support a classification scheme which can represent files as being organized in a hierarchy with a minimum of three levels.
- 3.1.3 The ERMS should not limit the number of levels in the classification scheme hierarchy.
- ...

3.2 Classes and Files

- 3.2.1 The ERMS must support metadata for files and classes in the classification scheme; and after a record has been captured the ERMS must restrict the ability to add to or amend its metadata to Administrators.
- 3.2.2 The ERMS must provide at least two naming mechanisms for electronic files and classes in the classification scheme:
  - a mechanism for allocating a structured numeric or alphanumeric reference code to each electronic file;
  - a mechanism to allocate a textual file title for each electronic file.

  It must be possible to apply both identifiers separately or together in the same application.
- ...

4.1 Access

- 4.1.1 The ERMS must allow the Administrator to limit access to records, files and metadata to specified users or user groups.
- 4.1.2 The ERMS must allow the Administrator to attach to the user profile attributes which determine the features, metadata fields, records or files to which the user has access. The attributes of the profile will:
  - prohibit access to the ERMS without an accepted authentication mechanism attributed to the user profile;
  - restrict user access to specific files or records;
  - restrict user access to specific classes of the classification scheme;
  - restrict user access according to the user's security clearance;
  - restrict users access to particular features (e.g. read, up-date and/or delete specific metadata fields);
  - deny access after a specified date;
  - allocate the user to a group or groups.

  An example of an accepted authentication mechanism is a password.
- ...

5.1 Retention Schedules

- 5.1.1 The ERMS must provide a function that specifies retention schedules, automates reporting and destruction actions, and provides integrated facilities for exporting records and metadata.
- 5.1.2 The ERMS must be able to restrict the setting up and changing of retention schedules to the Administrator.
- ...

Figure 1.2. Example requirements from **MoReq** Specification

As shown in the Figure 1.2 specification illustrates detailed features of e-documents. While there are 13 chapters, such as overview of ERMS requirements, classification scheme, control and security, retention and disposal, capturing records searching, retrieval and rendering, administrative functions, other functionality, non-functional requirements, metadata requirements, reference

model while all of these chapters give information about administration, access rules, controlling and etc /31/.

1.3. e-Government projects of different countries and role of electronic document exchange systems

As we said in the introduction part, paperless office leads to e-Government, while most of government services can be offered through network. As more private and some government offices starts exchange of e-documents, more part of society will be aware of advantages of such services, because there are lot of benefits both for service provider and user, Chapter 1.1 describe benefits of e-documents. As paperless office leads to e-Government and the widest implementation of e-document exchange is this project, in this Chapter we are going to discuss status of e-Government projects of some developed countries, which was carried by leading international e-Government project ranking Waseda University on 2008 April /32/.

The Waseda University World e-Government Ranking contains comprehensive benchmarking indicators in order to obtain an accurate and precise outcome for the latest development of e-Government in the world. These include: network preparedness, required interface-functioning applications, management optimization, homepage situation, introduction of CIO, and the promotion of e-government. This research does look into real operations, online services and the relationship between governments and their stakeholders.

In the public sector, more countries like the United States and Singapore are using various interactive tools to serve as tools to communicate with their citizens in terms of environment protection policy, regional recognition and even political issues with an interactive approach so that both government and citizens could reach a win-win situation.

A total of 34 surveyed countries/economies for this project, while the top ten countries/economies in the ranking are: (1) United States of America, (2)

Singapore, (3) Canada, (4) Korea, (5) Japan, (6) Hong Kong, (7) Australia, (8) Finland, (9) Sweden, and (10) Taiwan see Figure 1.3.

2008		2007		2006		2005	
1	USA	1	USA	1	USA	1	USA
2	Singapore	2	Singapore	2	Canada	2	Canada
3	Canada	3	Canada	3	Singapore	3	Singapore
4	Korea	4	Japan	4	Japan	4	Finland
5	Japan	4	Korea	5	Korea	5	Sweden
6	Hong Kong	6	Australia	6	Germany	6	Australia
7	Australia	7	Finland	7	Taiwan	7	Japan
8	Finland	8	Taiwan	8	Australia	8	Hong Kong
9	Sweden	9	UK	9	UK	9	Malaysia
9	Taiwan	10	Sweden	10	Finland	10	UK

Source: www.obi.giti.waseda.ac.jp/e_gov/

Figure 1.3. Comparison on the 1st, 2nd, 3rd, and 4th ranking result

In order to obtain the latest and most accurate information, experts attended international e-Government conferences, and visited governments and think-tanks in major countries/economies. Finally, discussions with international organizations such as the Asia Pacific Economic Cooperation (APEC), the Organization for Economic Co-operation and Development (OECD), the International Telecommunications Union (ITU) and the World Bank, were held.

Main Trends of e-Government by Indicators. 1. Network Preparedness – In the area of network preparedness, the major foundation for implementing e-government such as Internet users, Broadband users, Cellular phone users, PC users and Security system have been well established, while an increasing number of countries have already reached the upper level.

2. Required Interface-Functioning Applications. There has been major progress in the development of required interface applications for the promotion of e-Government in many countries. This is reflected in the results obtained by countries/economies such as Japan and Hong Kong, currently in the top four places in the field of interface functions and applications category as compared to

last year's ranking for the same category. Based on the obtained ratings, the top three spots in the required interface category for this year are occupied by the United States, Singapore and Canada respectively.

Also, countries ranked in the top two (1) Canada and (2) Australia are now on third and fourth places. It is also important to note that there are more countries catching up to countries that initially led e-government rankings in the past, hence the number of countries that are tied to the same rank. This implies a slowing down of e-government initiatives of these countries (developed countries) and the acceleration of e-government initiatives in more developing countries /32/.

In the area of e-government applications, e-Tax and e-Tender applications seem to be most widely implemented. E-Voting, on the other hand, is encountering legal issues in some countries resulting in the slowing down of initiatives in this field.

3. Management Optimization. A growing number of government organizations have realized the need to continuously review and revise their internal processes so as to capitalize on the advantages of ICT while at the same time deliver quality services to all stakeholders. Competition has been fierce this year and many countries have made vast improvement in their effort of optimizing management within their government and these have been reflected in the results for the Management Optimization indicator.

Singapore, Hong Kong, USA, Korea and Sweden have maintained their stride and continue to build on what they have achieved in the past year. As a new economic power for the 21st century, Indian government's effort in optimizing and integrating its public sector will be closely monitored.

4. Homepage. Homepage ranking focuses on four main areas: updating frequency, public disclosure, link navigation system, and multi-language correspondence. The top ten countries in the homepage ranking came from (1) Canada, (1) Hong Kong, (1) USA, (4) Korea, (4) Norway, (4) Sweden, (7) Australia, (7) Finland, (7) Japan, and (7) India, see Figure 1.4.

Network Preparedness	
1	Sweden
1	Netherlands
3	Singapore
4	USA
4	Norway
4	Australia
4	Finland
8	Japan
8	Canada
8	Germany
8	Hong Kong
8	New Zealand
8	Taiwan
8	UK

Interface Function and Applications	
1	USA
1	Singapore
3	Canada
4	Australia
4	Hong Kong
4	Korea
4	Japan
8	Taiwan
9	New Zealand
9	Sweden

Mgt. Optimization	
1	Norway
1	Singapore
3	Canada
3	Hong Kong
3	USA
3	Finland
7	UK
7	Korea
9	Sweden
9	Italy
9	New Zealand

Homepage	
1	Canada
1	Hong Kong
1	USA
4	Korea
4	Norway
4	Sweden
7	Australia
7	Finland
7	Japan
7	India

Introduction of CIO	
1	USA
1	Singapore
3	Canada
3	Japan
5	Korea
6	Australia
6	Germany
8	Hong Kong
8	South Africa
8	Taiwan
8	UK

Promotion of e-Gov	
1	Canada
1	USA
1	Singapore
1	Japan
5	Korea
5	Sweden
7	Italy
8	Norway
8	Finland
10	Australia
10	Hong Kong
10	Taiwan
10	India

Source: www.obi.giti.waseda.ac.jp/e_gov/

Figure 1.4. Top 10 Ranking for Each Sector

Most of the countries/economies that occupy the top ten fulfilled the above requirements. However, for multi-language correspondence, some of the countries (for example, New Zealand and Australia) do not have multi-language option in their homepages. As for countries/economies analyzed last year that did not have

multi-language homepages (the United Kingdom, the United States, the Philippines, and Singapore), only the United States and the United Kingdom have implemented this option in their homepages. As for the Philippines, the multi-language option has been partially implemented.

5. Introduction of Chief Information Officer (CIO). One each and also from the data gathered by the researchers, it shows a trend that the promotion of the CIO function comes after the other functions of the e-Government which focuses on implementations like Network Preparedness, Homepage, Interface Functions and Applications and lastly Promotion of e-Government. On the other hand CIO and management functions seem to receive lesser attention /32/.

6. Promotion of e-Government. More nations are increasing their efforts in e-government promotional activities. The United States is still in first place for its e-Government Promotion Activities (see Figure 1.5), sharing this position with Canada, Singapore and Japan, which have improved their scores for this field, compared to last year.

Rank	Country/Economy	Deviated Score		Rank	Country/Economy	Deviated Score
1	USA	68.3		18	Malaysia	49.4
2	Singapore	67.8		19	Netherlands	45.4
3	Canada	66.8		20	Thailand	44.9
4	Korea	63.2		21	Spain	44.3
5	Japan	62.2		22	Indonesia	43.8
6	Hong Kong	61.7		23	China	43.3
7	Australia	59.1		24	Philippines	42.8
8	Finland	58.6		24	Brazil	42.8
9	Sweden	56.6		26	Chile	42.3
9	Taiwan	56.6		26	South Africa	42.3
11	Italy	56.1		26	Mexico	42.3
12	England	55.6		29	India	41.8
13	Norway	55.1		30	Brunei	39.8
13	Germany	55.1		30	Vietnam	39.8
15	New Zealand	50.5		32	Russia	37.7
16	France	50.0		33	Peru	36.2
17	Belgium	50.0		34	Fuji	28.0

Source: www.obi.giti.waseda.ac.jp/e_gov/

Figure 1.5. 4th Waseda University World ranking e-Government 2008

Korea has dropped from second place to fifth. In a similar way, Finland and Australia have descended from second place to eighth and tenth respectively. Four countries/economies (Sweden, Norway, Hong Kong and Taiwan) have improved from last year to be part of the top ten in this category.

As we see on these figures world's most developing countries are vastly going to implements electronic form of their government, while this purpose can't be achieved without wide implementation of e-documents. Because purpose of e-Government is, providing wide rang of information and services distantly, besides going to the physical place of appropriate offices. While the most of people need some official document, e-Government implementation can't be achieved without fine grained e-document exchange system.

1.4. Purpose, scope and expected results of final qualifying work

Purpose of final qualifying work. Development rate and level of information technology depends on how much beneficial is this sphere for society and industry. Rapidly development and integration of computer technology leads to employment of these technologies to every branch of society.

Nowadays every developing country, in this case, Uzbekistan also tends to increase rate of information technology development. Electronic document exchange is also governmental and national level project, where its realization allows us to manage business and social processes faster, efficient and effectively. Besides this, not only higher social and economical outcome and positive changes are expected through applying electronic document exchange system, but also governmental management, control and manipulation of office documents are estimated to be less expensive, hence official municipalities will be able to provide low cost and high quality informational and other wide range services to companies and individual citizens.

Key point of electronic document exchange is – security solutions afforded in it. While, even we deal with small company which consists of a number of employers or large scale corporations or government agencies, each of them deal with some secure information. In the case of confidential information, system has to provide a number of options as tampering of information by the third part, non-repudiation of authors, protection from eavesdropper and etc.

The purpose and scope of this final qualifying work is discuss, analysis different cryptographic tools used to work with vital information and design solution for working with confidential information in electronic document exchange system.

Scope and expected result of final qualifying work. This final qualifying work devoted to one of the most important task of electronic commerce, which is electronic document exchange system and provide processing with confidential information in it.

In final qualifying work, all over information such as demand for electronic document exchange system, purpose and applying sphere in society and industry are covered first. Next, the widest application which uses electronic document exchange in large scale – e-Government projects of developed and developing countries are discussed by some criteria. Including the way of each country in its e-Government project, accomplished and planned tasks, ranked positions in international arena are given by years. The next part of this final qualifying work is devoted to international electronic document formats, structure, requirement and recommendations. To illustrate real structure, content and control way of electronic document, one of the most e-Government project leader country's proposals is analyzed. After being aware of overall status and conditions, different cryptographic tools which are used in confidential information exchange systems, like e-sign, encryption, hash function are discussed and architecture of electronic document exchange system which applies all of this cryptographic techniques is constructed. Next different cryptographic and technological solutions, which are used to provide higher security protection compared with each other. The rest of

work discusses applying sphere, future development of constructed and recommended solution.

Conclusion

In this chapter we discussed overall and current status of e-document exchange systems. First we saw actual demand and benefits from applying e-document exchange system in business and society. However, even in developed countries some obstacles are exists to fully implement e-document exchange system.

To develop technologies as a suitable for all business and government departments there was proposed standard structure, format and content for e-documents. Now widely available three such standard formats, first “MoREQ” was developed by IDA (Interchange of Data between Administrations) Programme of the European Commission, second “DoD 5015.02-STD, Electronic Records Management Software Applications Design Criteria Standard” by Department of Defense United States and last “JSR 170, Standardizing the Content Repository Interface” an open Source technology targeting worldwide usage.

As paperless office services increase and society moves to paperless, governments also will be able to provide its services over network, while this leads to build e-government of each country. Now e-Government project of each country is fulfilling continuously.

Concluding overall status and main fields of e-document exchange, purpose and scope of this final qualifying work is discuss, analysis different crypto graphical tools used to work with vital information and design solution for working with confidential information in electronic document exchange system.

II. Constructing structure of electronic document exchange system

As we saw in Chapter 1.2 there are different implementations of electronic document exchange systems. But e-document exchange system has to be well constructed, distinguishable and in easily controllable format (see Charter 1.2).

Of course, as document is well constructed, it will be easily stored and exchanged. In this chapter we illustrate structure of electronic document exchange systems, which will work as a ground for the rest of project.

In the Chapter 2.1 we illustrate real construction and working example of e-document, discussing its structure, content and management issues. The structure is recommended for multi-level electronic document management systems, such as government agencies or international companies. This is model, which put into operation in one of the e-Government project leaders, Taiwan.

Chapter 2.2 devoted to discussion of database scheme of electronic document exchange system, discussing 10 tables which are used to administration and storage of electronic documents. This chapter defines purpose and benefits of such electronic document exchange system's database structure.

Chapter 2.3 illustrates different cryptographic tools which were applied to confidentially exchange within vital information through public accessible network. Here we are going to discuss worldwide implemented crypto algorithms, which are used for electronic sign generating and verifying, encryption and archiving.

Chapter 2.4 devoted to discussion, realization and special properties of new hybrid cryptosystem, namely Private Box algorithm, which were constructed by the author within supervisor Rustam Khamdamov.

Cryptosystem calls hybrid because it uses symmetric algorithm for secure key agreement between two abonent and for the further information exchange,

these key will be used as a seed parameter for generating actual encryption keys. Chapter 2.4. provides detailed information and each encryption steps.

2.1. Constructing fine-grained electronic document exchange model

The global acceptance of electronic commerce and the progress made in network technologies have great impact on the development of electronic government (e-Government). One of the main objectives of e-Government is to exchange information between government agencies in a timely manner. From 2001 until 2004, one of the major applications of the e-Government program enabled the secure and effective exchange of official documents between governmental agencies. As a result, an exchange model has been developed. In the model, as illustrated in Figure 2.1, an official document is first transformed into XML format; then the XML-based document is signed and encrypted; and finally the encrypted document is sent to the Official Document Exchange Center. Next time when the recipient logs in, the ciphered document will be retrieved from the center. The received ciphered document will then be decrypted and verified in order to obtain the original document.

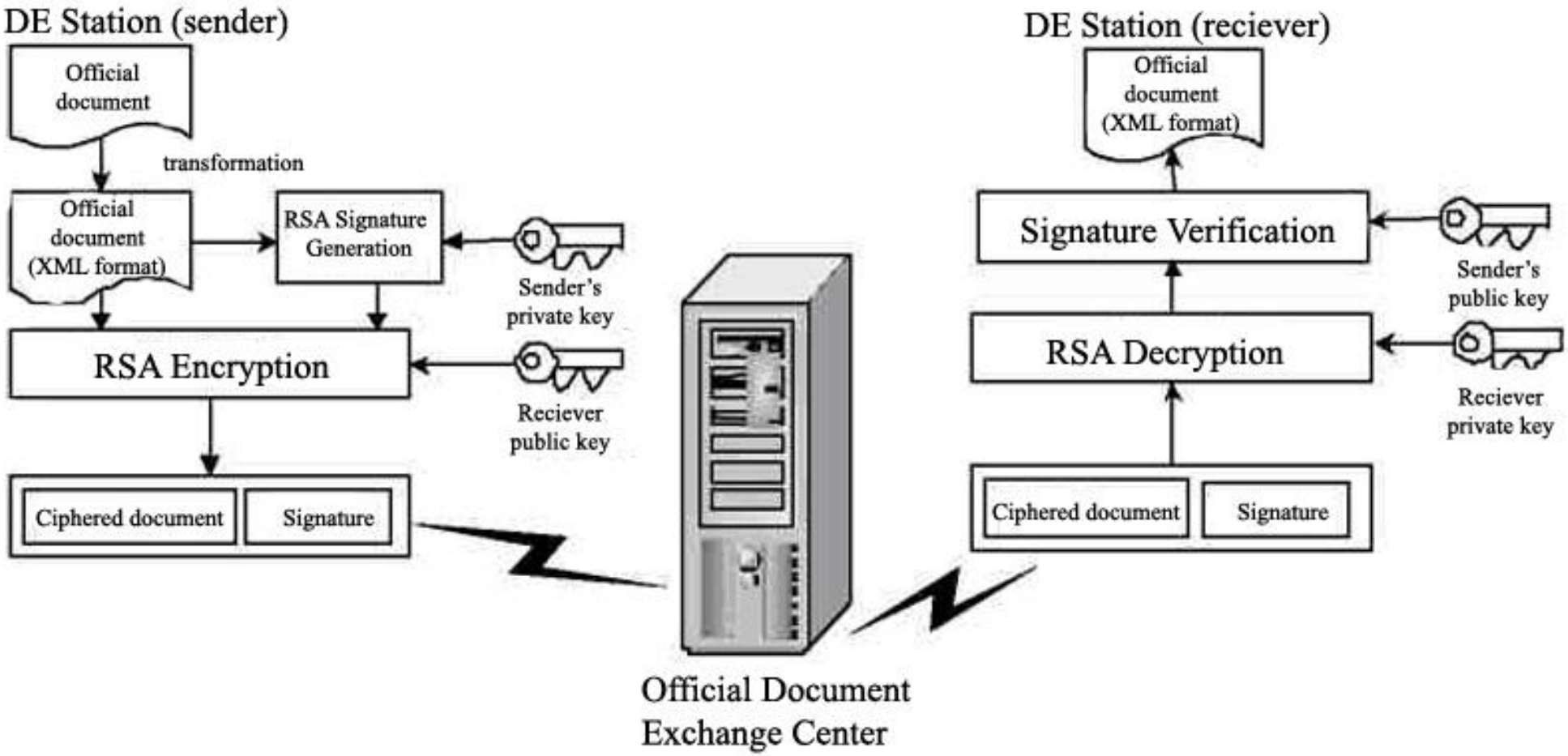


Figure 2.1. An Official Document Exchange Model

In the current implementation, each agency assigns a specialized staff to send and receive official documents. When an official document, classified as "confidential," is received, the staff is still able to decrypt the whole document and read its content even though s/he is not authorized. Therefore, it is very important to design a secure official document exchange model with fine-grained control so that only designated recipients are allowed to read and process the received documents, while the staff that receives the incoming documents is only allowed to verify their validity.

To achieve these objectives, it is required that the official document contains all the information needed for encryption, signature, and access control. Traditional security mechanisms, in view of the fact that the target of access control is usually the whole document, require more than one file to accomplish these tasks. Fortunately, due to the rich structure of XML, *it is easy to integrate encryption, signature, and access control into one XML document*. Therefore, a fine-grained official document exchange model for e-Government is proposed here. Taking advantage of the rich structure of XML, the content of the XML documents can be encrypted at various security levels /10/. Example of an official document is given in Figure 2.2. The official document has two parts – header information and content of the document. The header indicates that the document was issued on 2004/09/30, the official document is classified as "confidential," and the recipient is the Ministry of Defense. The body of the document describes the budget that is required to purchase missiles to enhance national security. The staff at the receiving desk will be able to read the header information of the received document. However, since the arriving document is classified as "confidential" s/he will not be able to read the content of the document. Instead, only the designated staff is allowed to decrypt and read the document.

```
<officialDocument>
 <headerInfo>
    <issueDate>2004/9/30</issueDate>
    <securityLevel> Confidential</securityLevel>
    <receiver>Ministry of Defence</receiver>
 </headerInfo>

 <content>
    <subject>Procurement of missiles</subject>
    <description>
       To enhance national security
    </description>
    <budget>250000000</budget>
 </content>
</officialDocument>
```

Figure 2.2. An Official Document Example

The content of an official document can be encrypted by different keys based on the security level. And the staff can only decrypt and read the content of the document if s/he is authorized. In addition, due to the fact that the RSA cryptosystem is adopted in the proposed model, it is secure and can be easily incorporated into existing implementations.

A Smartcard-Based Framework for Secure Document Exchange. A secure document exchange model was proposed by Yang, Ju, and Rao. There are many Document Exchange (DE) Stations and a Security Management Center in the model, as shown in Figure 2.3. Each DE Station is responsible for sending and receiving documents and equipped with a smart card reader. Cryptographic algorithms are embedded in the smart card to provide security functions for digital signature and digital envelope. The Security Management Center is responsible for issuing smart cards, acting as a public-key certificate authority, regularly publishing certificate revocation list, and maintaining distribution lists. A distribution list contains a group of recipients that can be used as a mailing list to send and receive official documents.

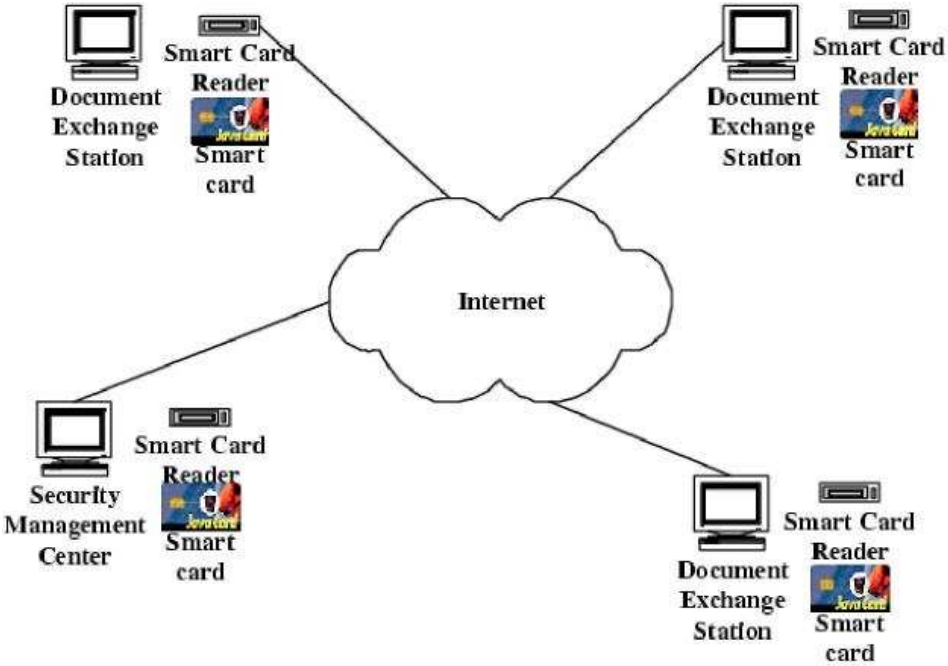


Figure 2.3. Yang-Ju-Rao's Secure Document Exchange Model

An official document is first transformed into XML format, and the XML-based document is then signed and encrypted. Upon receiving the ciphered document, the recipient decrypts, verifies, and obtains the original document.

XML Signature. For ensuring the authenticity and the integrity of the data transmitted over the Internet, digital signature techniques are widely adopted. If the signature is validated, the transmitted document is integral and authentic. The digital signature is based on public-key cryptography in conjunction with one-way hash functions. After creating a one-way hash value (called message digest) from the document, the signer encrypts the digest value with her/his private key. In the traditional signature techniques, every participant signer signs the whole document rather than these portions of the document that s/he is responsible for. This brings two major drawbacks. One is that it requires extra communication cost to transfer the whole document and extra computation time to generate personal signatures on the whole document. The other drawback is that it is difficult to achieve the principle of responsibility separation. It is extremely time consuming and tedious to read the whole document before signing it. In practice, every signer should sign these parts of the document that s/he is responsible for. To overcome these

drawbacks, Lu and Chen proposed a novel XML multi-signature scheme in 2003 that provides fine-grained control at element level.

XML Encryption. Various encryption techniques have been designed for encrypting a whole document, but they do not support selective encryption of a document. However, a requirement of many applications is that users have the ability to encrypt only selected portions within a document and encrypting different portions of the same document with different encryption keys. To meet this requirement, XML Encryption has been proposed by the W3C XML Encryption Working Group /15/.

An XML Encryption structure is composed of two parts and they are `EncryptedInfos` and `Objects`, as is illustrated in Figure 2.4. The information needed for a correct decryption is stored in the `EncryptedInfos` element, and the encrypted data are contained in the `Object` element.

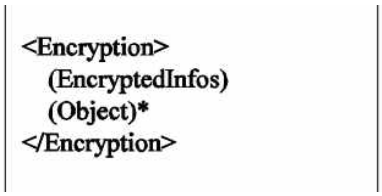


Figure 2.4. Structure of the XML Encryption

Design of a Fine-Grained Official Document Exchange Model. The basic idea of the proposed model is that the originators or senders can sign and encrypt selective portions of a document with different keys based on the security policies. Therefore, only the designated recipients are allowed to read and process the received documents.

The proposed secure fine-grained official document exchange model is shown in Figure 2.5. The model includes one sender side, one official document exchange center, and more than one receiving sides. First, each recipient has to register at the official document exchange center and the registration information is stored in a database. The registration information includes at least the recipient's name and her/his public-key certificate. Also, the role of the recipient in the government agency has to be registered. Additionally, with the roles employed in

the proposed model, the management of recipients (or government agencies) is much easier.

XML has become a standard format for data interchange on the web, and it is widely adopted by the governments to exchange official documents. Therefore, the development of a secure fine-grained official document exchange model is extremely important. Since the traditional document exchange models lack granular control on official documents, security leaks are possible. This article—taking advantage of the rich structure of XML, XML Signature, and XML Encryption — proposes a secure fine-grained official document exchange model for e-Government. Due to the fact that the RSA cryptosystem is also adopted in the proposed model, it is secure. Also, since the proposed model follows the XML Signature and XML Encryption specification drafts, it can be easily incorporated into existing implementations.

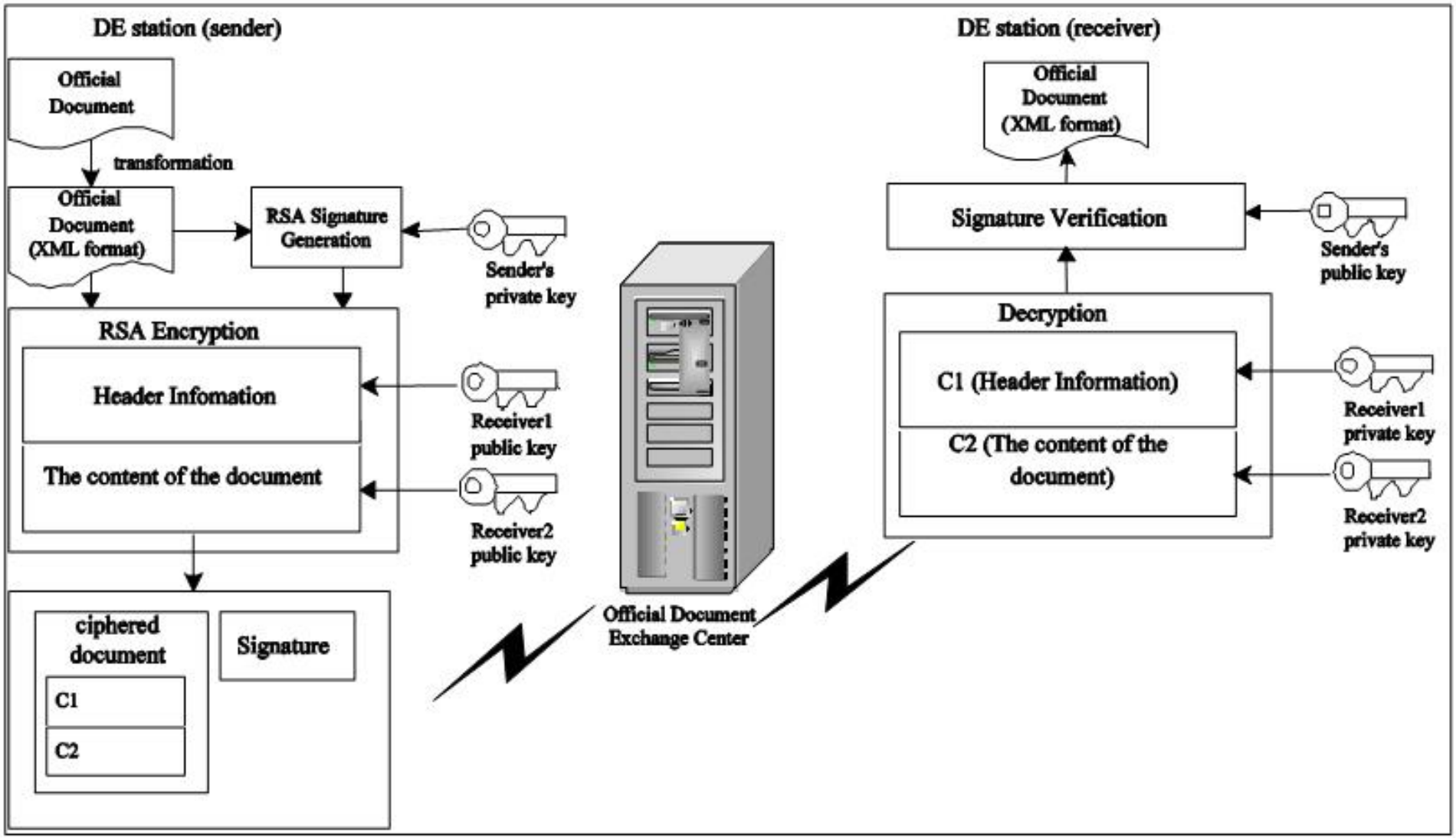


Figure 2.5. Secure Fine-Grained Official Document Exchange Model

2.2. Designing architecture of electronic document exchange system

Usually architecture of each system designs according to its functionality, purpose and scope of solution and to available libraries and technology. As we said in Chapter 1.4 our purpose is – organization of working with confidential information in e-document exchange system. Actually, this task consists of a number of subtasks, where the first part of work is, whole database system has to be created to store users' and administrators' record, and business logic of solution, i.e. each step of procedure has to be defined. In this Chapter we will see working algorithms of solution called "e-doc". Next, we will discuss database structure of electronic documents storage. Detailed manual with illustrated pictures will be given in the Chapter 3.1.

System's working algorithm. In the Figure 2.6 below we illustrated each steps of users' as a programming algorithm, to make use of system. First we will see overall scheme of working algorithm, while detailed steps will be shown in next Figures.

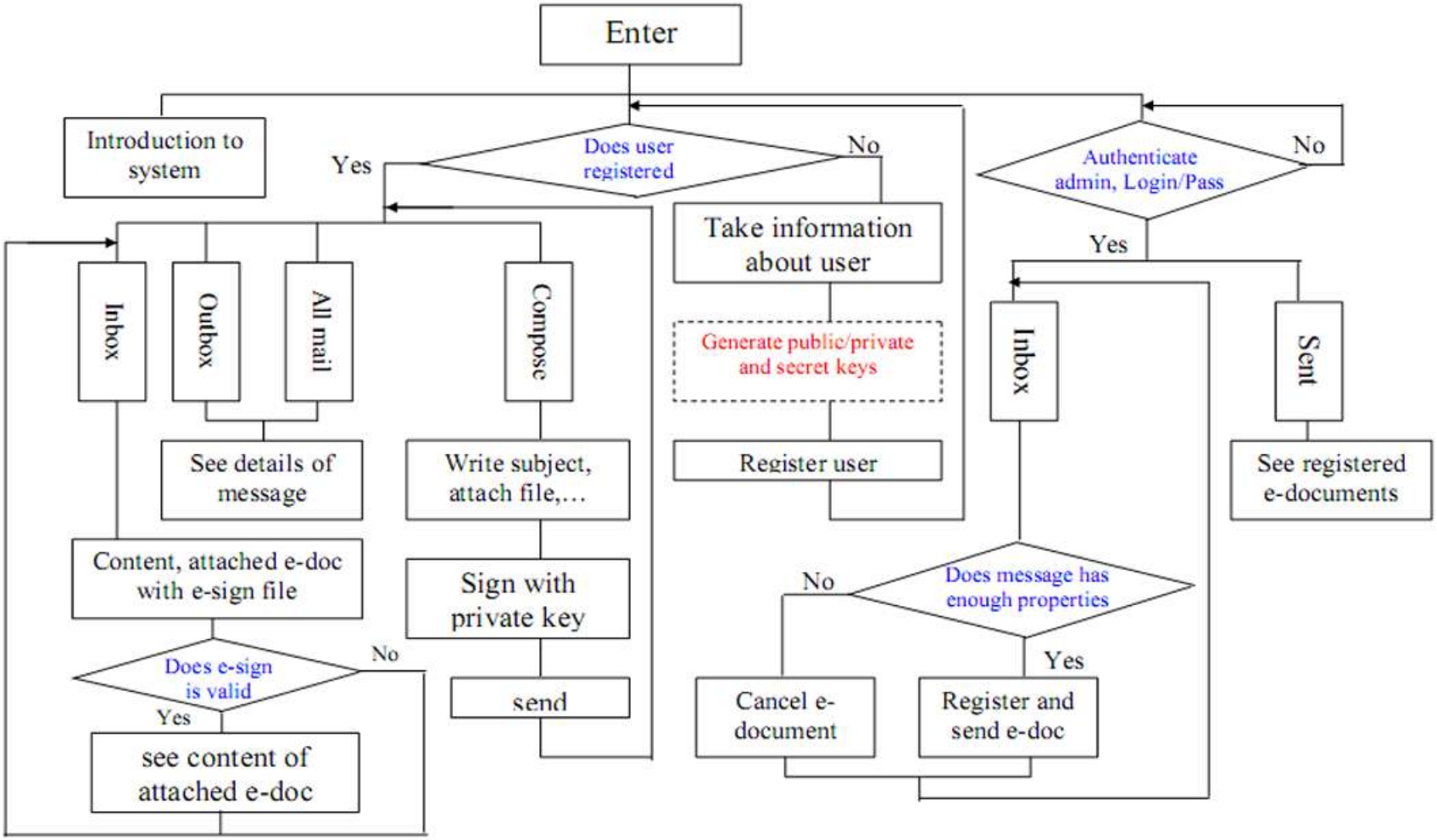


Figure 2.6. Overview of system working algorithm

As here described in Figure 2.7, user enters to welcome page, where mission and vision of this project is explained. Here, user has two other choices, where first options is only for e-document office managers, which requires Login/Password authentication to login into system and second choice is for users, where only pre-registered users can exchange e-document with the help of system or with sign up option for users, who is not pre-registered, but has demand for exchange e-documents via this system, depending on implementation of project s/he can be employer of some company or member of some group.

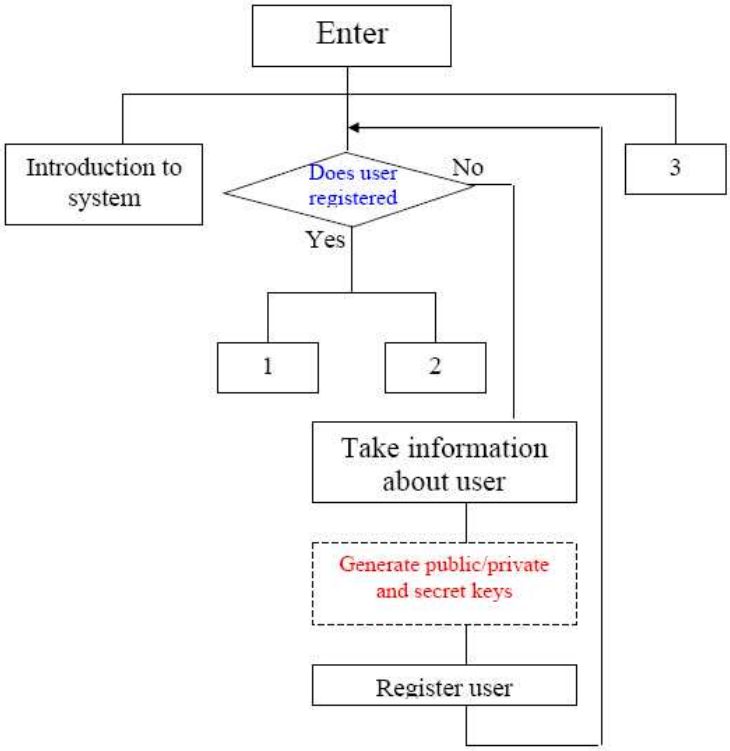


Figure 2.7. Algorithm of e-document exchange system

As whole algorithm is difficult to show on one picture we divided it to four separate figures, whilst each of them illustrates some logical part. Next we'll explain each of them in Figures 2.8, 2.9 and 2.10.

Let's first discuss case of users'. When a user clicks *Login to System*, he will be asked for a login and password, if he had been registered before, if not, user can easily sign up choosing *Sign up* option. During registration user will be asked for some information about him/herself, for instance first name, last name, surname, secondary e-mail, phone, physical address and etc. At the same time user has to

choose login and password for further usage of system. As user creates account, he will be able to login via previously given login and password. Here user can exchange electronic documents only with other users of same system.

At the backend server, during registration system automatically generates unique public/private and secret key pairs to sign electronic document.

Compose option of system allows users to compose messages and attach electronic documents to his/her mail, signing it with either with private or symmetric secret keys Figure 2.8. As in a real document management system, composed mail will not be delivered to destination without registration of central office. So, sent message automatically comes to office and office manager will be prompted to registration of new message with unique ID, where it immobilizes users from repudiation.

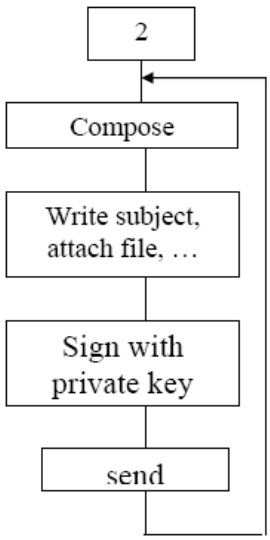


Figure 2.8. Compose e-mail with attached e-document

Here office manager checks only general options of message, like contains appropriate subject, author and etc. However, office manager has not any access to the content of the message or to the attached electronic document, while manager only checks overall properties of the message. Where his/her duty is only register incoming message, document and deliver it to destination Figure 2.9.

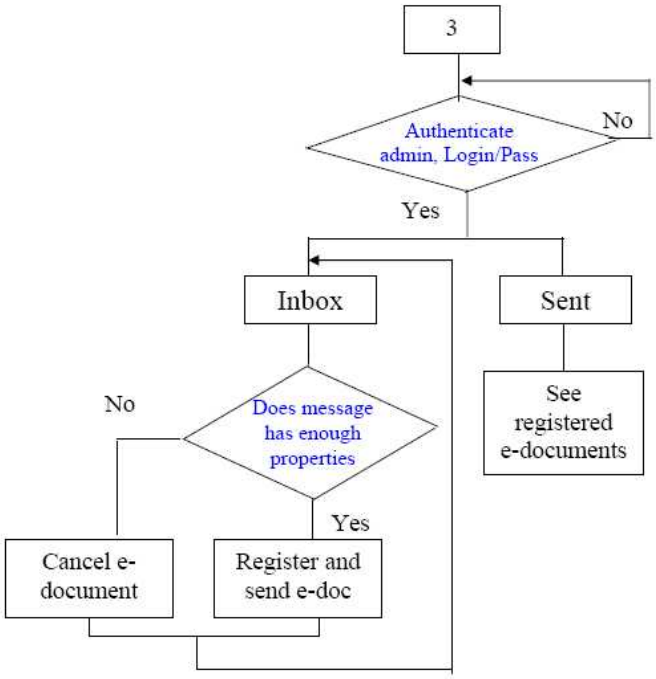


Figure 2.9. e-Document management steps by office manager

After registration of e-document in office manager will receive incoming e-mail with author signed document and e-document itself. When user opens message, he will be able to read content of the message with registered ID, date, sent date, and subject, while they will be shown as a message properties. But, before observe attached electronic document, it has to be validated, whether if document verification is approved, it will be shown to user, on the other case document will not be presented to user just giving appropriate alert, that document content was modified or e-sign was tampered during deliver Figure 2.10.

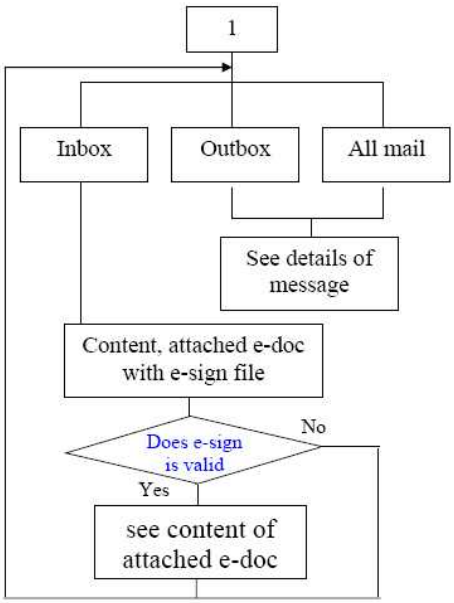


Figure 2.10. Incoming message of user

Of course, one can say that what is benefit from just not showing document? Does system can't provide better management? As, if e-sign validation is not approved system has to be inform registration center or author of document about this issue. However, as we mentioned previously in Chapter 1.4 (Purpose, scope and expected results of final qualifying work), purpose of this project is enabling process with confidential information in electronic document exchange system, while false validation of document allows us detect that confidentiality of e-document was corrupted, hence we can do further management of electronic documents and e-mail, taking in account in what environment system was implemented and what kind of management issues has to be done, such as informing registration center, sending report to author and etc. While this features of the system is are out of scope of this project's scope and system enables verification of e-documents for confidentiality, and accomplishes it with e-sign verification. As we'll see in Chapter 3.1 (Planning electronic-document exchange software, environment and confidential document flow) this option is already applied feature, checking e-sign of e-document.

Database architecture. As we explained earlier in this chapter, users will exchange e-mails, which contain e-document and e-sign. As we illustrated in Figure 2.7, all functionality of the system has to be supported by database storage.

In this chapter we will show only database architecture of system, while different RDBMS' (Relationship Database Management System) features, advantages will be shown and discusses in Chapter 3.1.

Database contains 10 tables, which has relationship with each other, with the primary and foreign keys. Figure 2.11 shows relationship of each table with each other.

There are total 10 tables, where seven of them is used for e-document management and remained three tables are used to store user accounts and central office administrator Figure 2.12.

Table *all_mail* is used to register each incoming and outgoing messages, while each e-mail contains attached and signed e-document, there is *filelocation* and *signlocation* columns. As we said previously in this chapter, users will exchange e-documents with each other, but only through central registration centre. So, before sending e-mail to that centre all properties of e-mail and e-document has to be recorded in *all_mail* table, whilst *my_user_to* defines recipient user to whom *my_user_from* sends e-document. While *my_subject* and *content* columns contains subject and composed content of message, *my_date* column is used to store composed and sent date of this message. *sign_algo, encrypt_algo, zip_algo* column describe cryptographic tools which were used to apply the security of e-document. Distinguishing from other columns *reg_id* and *reg_date* firstly fills with *null* where this value will take its significance after registration of e-document by central registering office. The last column *my_status* has foreign key *status_mes* table where each state of message has unique integer value and *status_mes* table describes all code of message state. As integer value is better to store and manipulate this technique is used in other tables too.

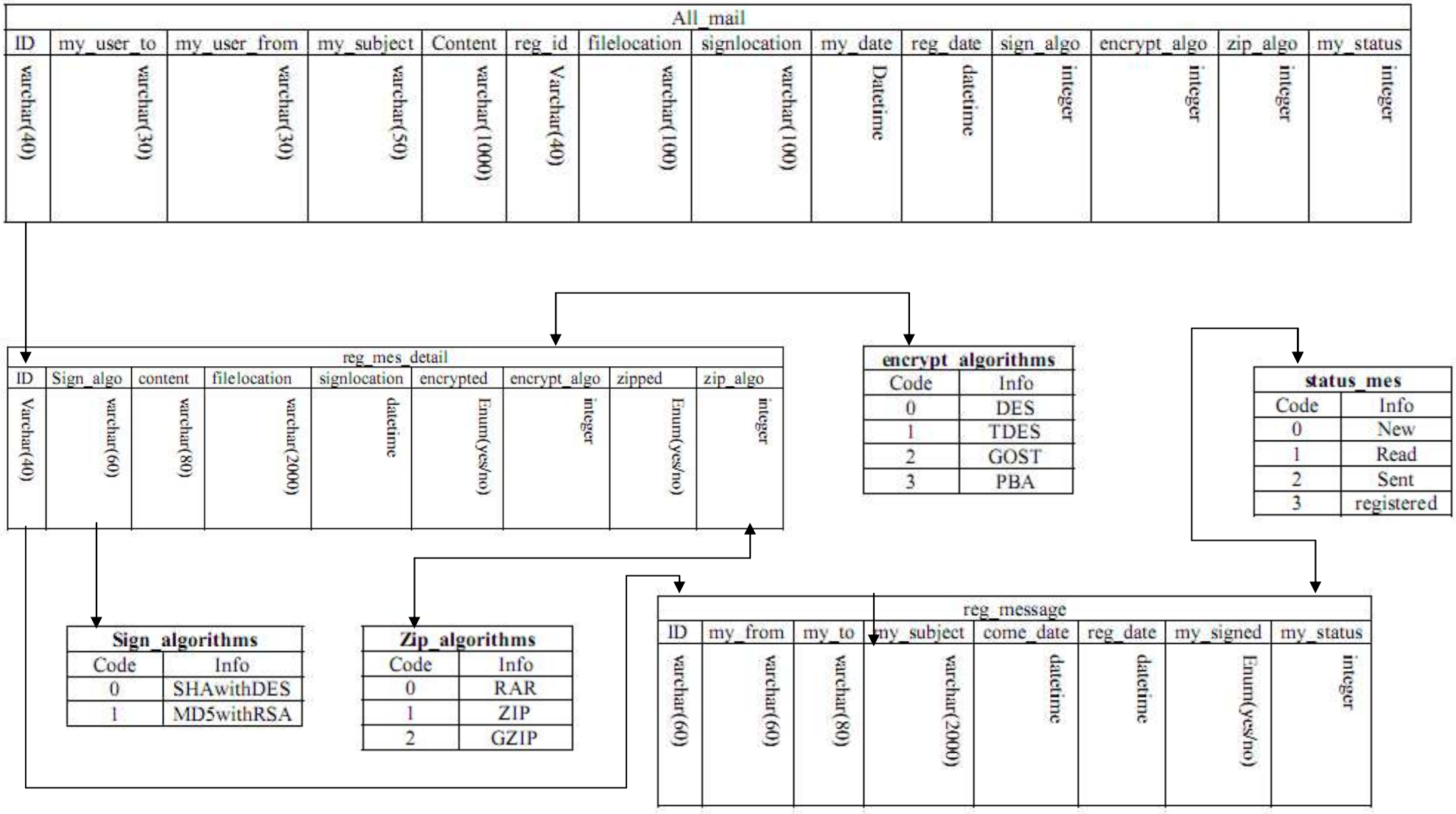


Figure 2.11. Relationship of tables for e-document exchange

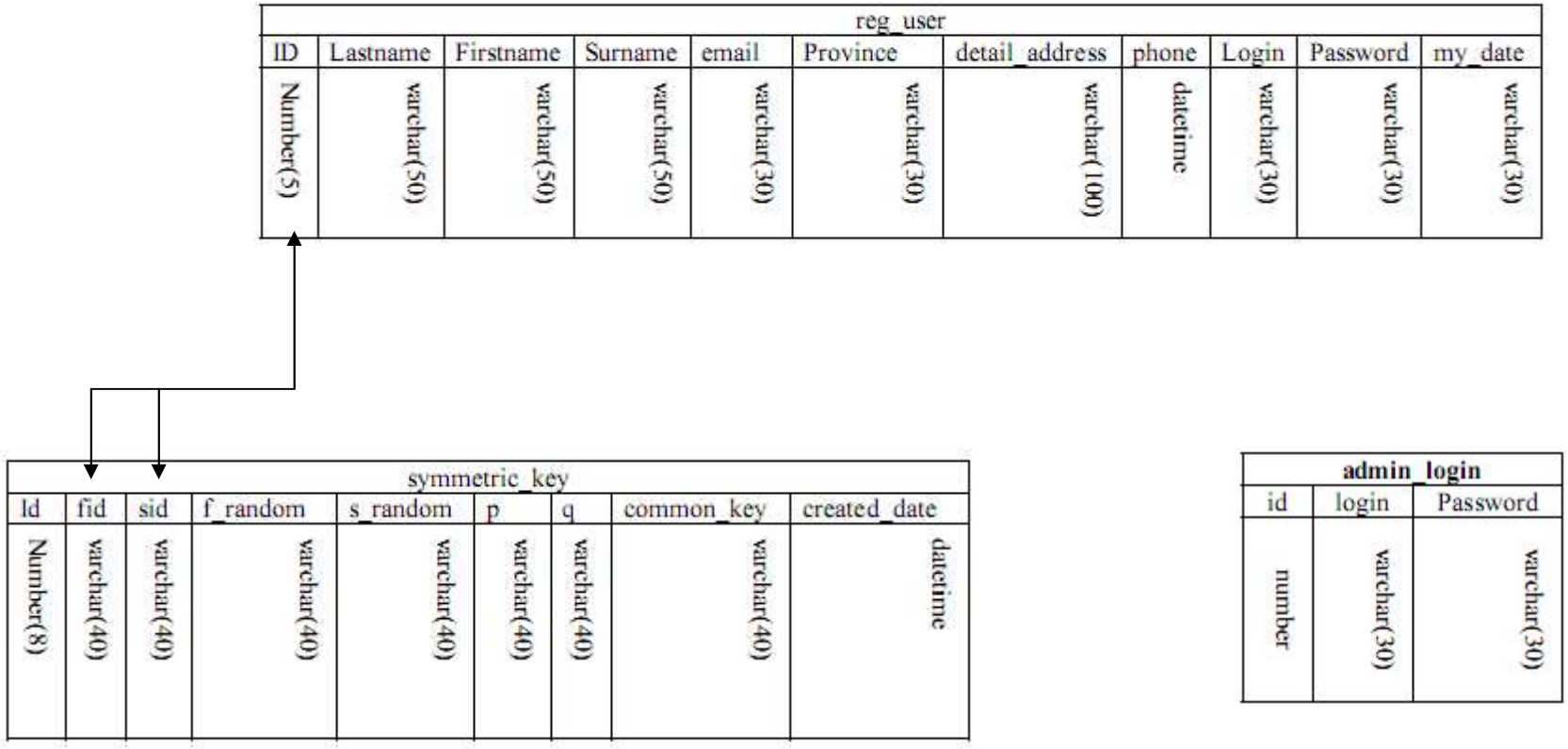


Figure 2.12. Tables of database to manage user and manager accounts

Such as *encrypt_algorithms*, *sign_algorithms* and in *zip_algorithms* tables, however all of these tables have only two columns, first called *code* which defines integer number used instead of string in that tables and *info* column which represents string explanation of that code. Anytime we are able to take string representation of code, referring to the second column of table. Additional comfortable feature of such coding is, it is easy to extent encryption, e-sign and archive algorithms, just adding one more row in these tables and using its code in other different tables.

Tables which are used by central office managers are *reg_message* and *reg_message_detail* tables where each of them stores status and detailed information of all exchanged e-documents.

In the Figure 2.12 users' and administrators' accounts are stored. At the time of user registration signing up user has to give some more information about him/herself where this information is used to manipulate e-documents in case of irresponsibility and for non-repudiation. As shown in the first table user has to give his/her first name, last name, surname, province, phone number and complete information about physical address. Secondary e-mail address is used to keep in touch in emergency cases, such as user forgets login/password, password knowing login or if first user want impersonate knowing friend's login and trying different pass phrases as the first's password, second user will be automatically prompted about this event through secondary e-mail.

Table *symmetric_key* is used to store secret keys, generated with Diffie-Hellman secret key agreement algorithms, enabling users to sign e-documents via their secret keys, of course in future development we'll make public/private keys generation and storage table, but this depends on real requirement to such e-document exchange systems. Here *fid* and *sid* is columns defines secret key pair between these to first and second user.

Table *admin_login* simple stores administrators' login and password, and additional opportunity to management, in case of more than one of central e-document managers.

2.3. Applied cryptographic tools on working with confidential information in electronic-document exchange

Security options of digital information can be gained either appliance based or as software. Nowadays there exists third type of digital information protection, calls as a middleware, which is integration of software and hardware. Both choices hardware (HW) and software (SW) have its own advantages and disadvantages, usually hardware provide better protection, they are not so portable, extensible and easy to implement as SW.

In this project I didn't use any specific hardware but here I used new cryptographic algorithms which were constructed and tested by author with supervisor, see Chapter 2.4.

In SW based protection the most important thing is crypto algorithm, while it defines all process of encryption and decryption, it is easy to develop as software and implement in web application, operation system or RDBMS.

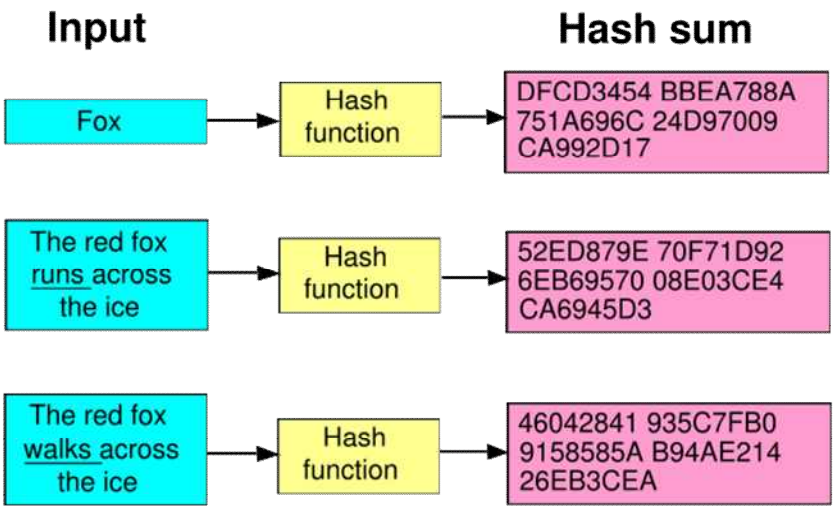
Here I divided cryptographic algorithms to four groups, whilst one of them is not actually cryptographic tool, but mixed form of other two, on the other hand as they used them to accomplishment of different purposes, I divided them in more detailed pieces.

Hash functions. Cryptographic hash function is a transformation that takes an input and returns a fixed-size string, which is called the hash value. Hash functions with this property are used for a variety of computational purposes, including cryptography. The hash value is a concise representation of the longer message or document from which it was computed. The message digest is a sort of "digital fingerprint" of the larger document. Cryptographic hash functions are used to do message integrity checks and digital signatures in various information security applications, such as authentication and message integrity.

A hash function takes a string (or 'message') of any length as input and produces a fixed length string as output, sometimes termed a message digest or a digital fingerprint. A hash value (also called a "digest" or a "checksum") is a kind

of "signature" for a stream of data that represents the contents Figure 2.13. One analogy that explains the role of the hash function would be the "tamper-evident" seals used on a software package.

In various standards and applications, the two most-commonly used hash functions are MD5 and SHA-1. In 2005, security flaws were identified in both algorithms /26/.



Source www.wikipedia.org

Figure 2.13. Hash function at work.

Hash functions has a lot of properties and criteria's which should be accomplished to provide better security, however there is no formal definition which captures all of the properties considered desirable for a cryptographic hash function. These properties below are generally considered prerequisites:

- **Preimage resistant:** given *h* it should be hard to find any *m* such that *h = hash(m)*.
- **Second preimage resistant**: given an input *m1*, it should be hard to find another input, *m2* (not equal to *m1*) such that *hash(m1) = hash(m2)*.

This property is implied by collision-resistance. Second preimage resistance is sometimes referred to as weak collision resistance.

- **Collision-resistant**: it should be hard to find two different messages *m1* and *m2* such that *hash(m1) = hash(m2)*. Due to a possible *birthday attack*, this means the hash function output must be at least twice as large as what is

required for *preimage-resistance*. This property is sometimes referred to as *strong collision resistance*.

In the Table 2.1 below we describe most popular and wide used cryptographic hash functions, with overall properties which were analyzed during exploitation.

Table 2.1. Properties of different cryptographic hash functions

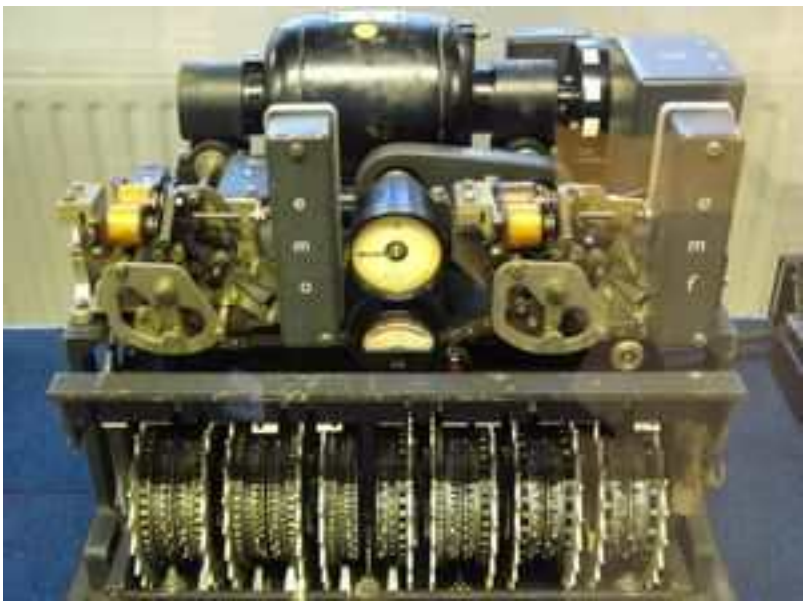
Algorithm	Output size (bits)	Internal state size	Block size	Length size	Word size	Collision
HAVAL	256/224/192/160/128	256	1024	64	32	Yes
MD2	128	384	128	No	8	Almost
MD4	128	128	512	64	32	Yes
MD5	128	128	512	64	32	Yes
PANAMA	256	8736	256	No	32	Yes
RadioGatún	Arbitrarily long	58 words	3 words	No	1-64	No
RIPEMD	128	128	512	64	32	Yes
RIPEMD-128/256	128/256	128/256	512	64	32	No
RIPEMD-160/320	160/320	160/320	512	64	32	No
SHA-0	160	160	512	64	32	Yes
SHA-1	160	160	512	64	32	With flaws
SHA-256/224	256/224	256	512	64	32	No
SHA-512/384	512/384	512	1024	128	64	No
Tiger(2)-192/160/128	192/160/128	192	512	64	64	No
WHIRLPOOL	512	512	512	256	8	No

Source www.wikipedia.org

Encryption. Encryption is the process of transforming information (plaintext) using an algorithm (cipher) to make it unreadable to anyone except those possessing special knowledge, usually referred to as a *key*. The result of the process is encrypted information (referred to as *ciphertext*). Decryption is reverse process to make the *encrypted* information *readable* again (i.e. to make it *unencrypted*).

Encryption, by itself, can protect the confidentiality of messages, but other techniques are still needed to protect the integrity and authenticity of a message; for example, verification of a *message authentication code*(MAC) or a *digital signature*. Standards and cryptographic software and hardware to perform

encryption are widely available, but successfully using encryption to ensure security may be a challenging problem.



Source www.wikipedia.org

Figure 2.14. The German Lorenz cipher machine

Figure 2.14 shows The German Lorenz cipher machine, which were widely used during WWII and there are other number of encryption machines, as *Enigma machine*, were used by militaries and governments to facilitate secret communication.

Digital signature. A digital signature or digital signature scheme is a type of symmetric or asymmetric cryptography used to simulate the security properties of a handwritten signature on paper. Digital signature schemes normally give two algorithms, one for signing which involves the user's secret or private key, and one for verifying signatures which involves the user's public key. However, normally symmetric cryptography digital signatures are not used in productivity, they are less famous. The output of the signature process is called the "*digital signature*".

A signature provides authentication of a *message*. Messages may be anything, from electronic mail to a contract, or even a message sent in a more complicated cryptographic protocol. Digital signatures are used to create public key infrastructure (PKI) schemes in which a user's public key (whether for public-key encryption, digital signatures, or any other purpose) is tied to a user by a digital identity certificate issued by a certificate authority. PKI schemes attempt to

unbreakably bind user information (name, address, phone number, etc.) to a public key, so that public keys can be used as a form of identification.

Of course this public key signature scheme is more secure then secret key cryptography's digital sign, however here all users has to use Public Key Infrastructure (PKI) to verify each other's public key. In case of symmetric digital signature users needn't to service of PKI and no certificate is required, but the most uncomfortable thing is users' agreement to secret key. Because some time users want to directly exchange document without secret key agreement.

Data compression. *Data compression* or *source coding* is the process of encoding information using fewer bits than an *unencoded* representation would use through use of specific encoding schemes. One popular instance of compression with which many computer users are familiar is the ZIP file format, which, as well as providing compression, acts as an *archiver*, storing many source files in a single destination output file.

As with any communication, compressed data communication only works when both the sender and receiver of the information understand the encoding scheme. For example, this text makes sense only if the receiver understands that it is intended to be interpreted as characters representing the English language. Similarly, compressed data can only be understood if the decoding method is known by the receiver.

Compression is useful because it helps reduce the consumption of expensive resources, such as hard disk space or transmission bandwidth. On the downside, compressed data must be decompressed to be used, and this extra processing may be detrimental to some applications.

The design of data compression schemes involves trade-offs among various factors, including the degree of compression, the amount of distortion introduced (if using a lossy compression scheme), and the computational resources required to compress and uncompress the data.

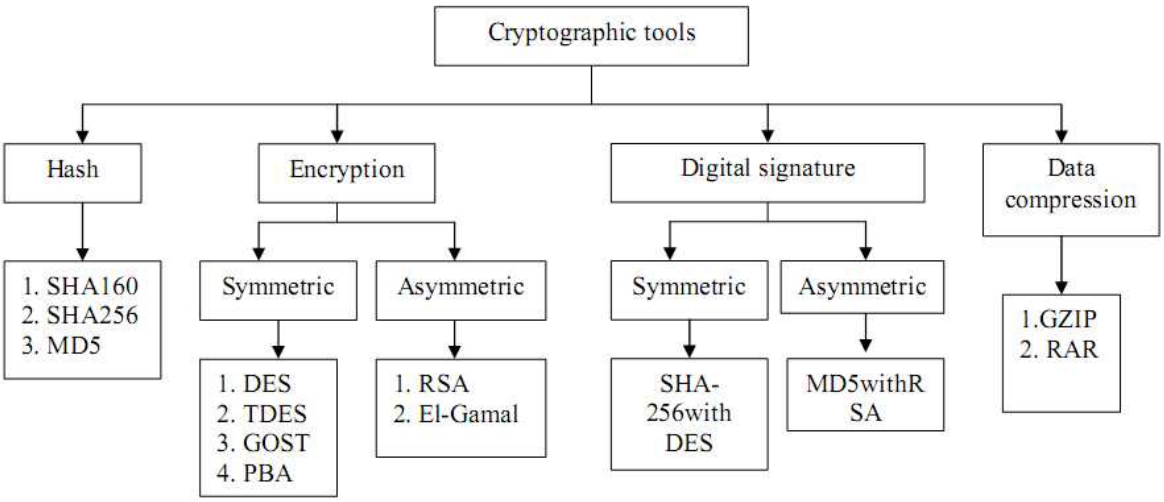


Figure 2.15. Applied cryptographic tools in e-document exchange system

Figure 2.15 below shows different cryptographic tools used to accomplish confidentiality of e-document exchange system. While there is rich library may be used for this purpose, but since we giving just example, and different algorithms can be used depending on environment and level of e-document confidentiality. For instance, as document is more secure, user can choose more robust crypto algorithm. Comparison of cryptographic algorithms in different criteria's given in Chapter 3.3.

Scheme described below was used as digital signature creation Figure 2.16.

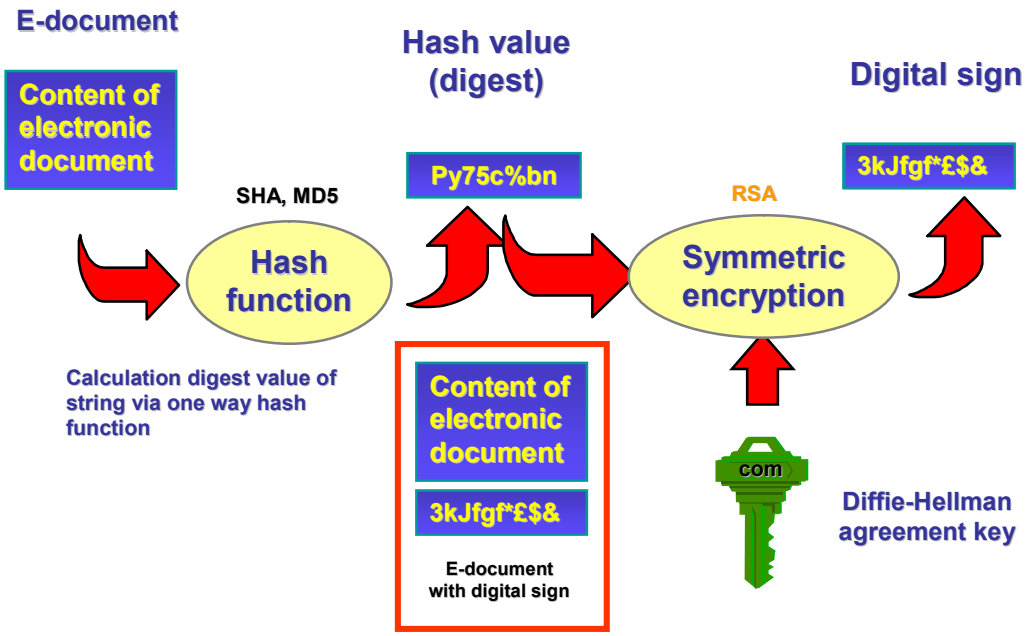


Figure 2.16. Digital sign generation

Two users agrees to common key via secret key agreement protocol, for instance Diffie-Hellman, while this key will be used to encrypt digest value of e-document. Sender sends same e-document in first file and encrypted message in second file to the recipient.

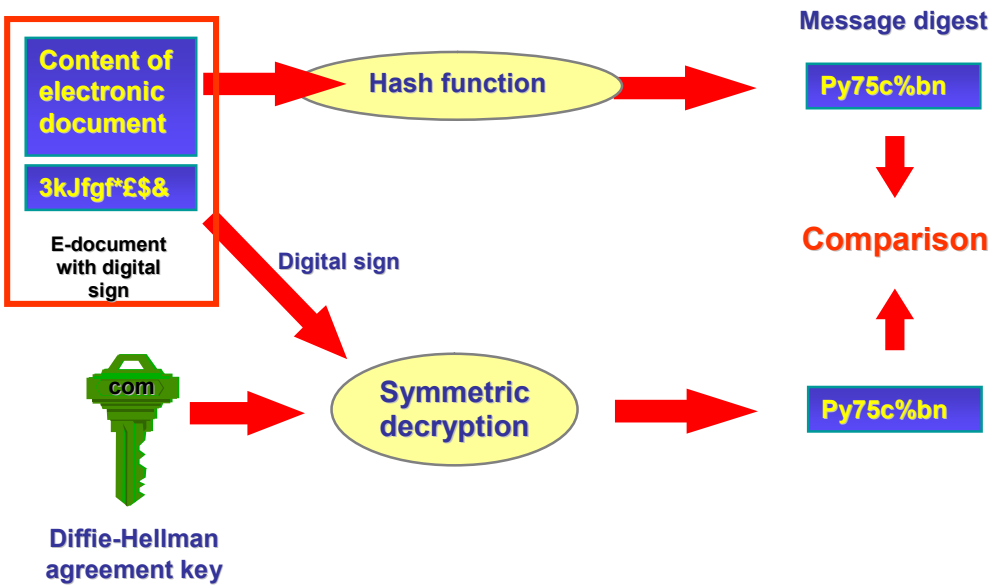


Figure 2.17. Digital sign verification

Digital sign verification can be done using e-document and encrypted digest value of message, while firstly recipient gives digest value of e-document and decrypting content of second encrypted file obtains digest value of original e-document. Comparing digest values of original (decrypted) and received e-document user verifies digital signature. Where if both digest values are same it means content of e-document and author is not modified, hence digital signature is valid on the other hand if digest values are not same digital signature fails on verification.

2.4. Applying new cryptographic algorithm in working with confidential information

Private box algorithm based on rucksack problems, which were first introduced in 1979 by Ralph Marklin and Martin Hellmann. That algorithm was one of algorithms which used rucksack systems and exchanged by elements through public channel after using modular arithmetic calculation for each element /6/. In suggested algorithm rucksack elements generate separately by each abonent on the basis of private parameter /11/. The name «*private box algorithm»* is followed from it. *Pseudorandom* numbers are used to generate private keys, which have predefined pattern /10/.

Private Box Algorithm. Assume, *A* and *B* abonents want to exchange with messages. Abonent *A* is – sender, and abonent *B* – is receiver here. To exchange by messages they follow these calculating steps.

1) Both abonents generate common private parameter $e_A^Z = e_B^Z = e^Z$ via some kind of key generating algorithms, such as Diffie-Hellman.

2) For ciphering open message $X = \{x_1, x_2, \ldots, x_l\}$, which contains letters of Z-alphabet, abonent *B* generates n - elements of private box $K^Z = \{k_1^Z, k_2^Z, \ldots, k_n^Z\}$ on the based on private parameter e^Z, using generator of *superincreasing pseudorandom numbers*[1]. After it codes of letters of open text - $X = \{x_1, x_2, \ldots, x_l\}$ will be taken, where length of each binary code equals to d. Merging binary code of characters to one sequence and divide it into m blocks we have *modified open text*,

$$X' = \{ x'_{11}, x'_{12}, \ldots, x'_{1n} \quad x'_{21}, x'_{22}, \ldots, x'_{2n} \quad \ldots \quad x'_{m1}, x'_{m2}, \ldots, x'_{mn} \}$$

where each block has length n. By scalar multiplication of modified open text X' and K^Z we take integer cipher $S = \{s_1, s_2, \ldots, s_m\}$ /11/.

[1] Method of generating described below

$$X^{'} = \{1 \quad 1 \quad ... \quad 0 \quad 1 \quad 0 \quad ... \quad 1 \quad ... \quad 0 \quad 1 \quad ... \quad 1\}$$
$$X^{'} = \{x_{11}^{'} \quad x_{12}^{'} \quad ... \quad x_{1n}^{'} \quad x_{21}^{'} \quad x_{22}^{'} \quad ... \quad x_{2n}^{'} \quad ... \quad x_{m1}^{'} \quad x_{m2}^{'} \quad ... \quad x_{mn}^{'}\}$$
$$K^{Z} = \{k_1^{Z} \quad k_2^{Z} \quad ... \quad k_n^{Z} \quad k_1^{Z} \quad k_2^{Z} \quad ... \quad k_n^{Z} \quad ... \quad k_1^{Z} \quad k_2^{Z} \quad ... \quad k_n^{Z}\}$$
$$S = \{x_{11}^{'} \cdot e_1 + ... + x_{1n}^{'} \cdot e_n = s_1 \quad x_{21}^{'} \cdot e_1 + ... + x_{2n}^{'} \cdot e_n = s_2 \quad ... \quad x_{m1}^{'} \cdot e_1 + ... + x_{mn}^{'} \cdot e_n = s_m\}$$

And we send it to *A* abonent.

3) Maintaining foregoing scheme abonent *A* also generates n elements of superincreasing private box $K^{Z} = \{k_1^{Z}, k_2^{Z}, ..., k_n^{Z}\}$ via his private parameter e_A^{Z}.

To obtain modified open text

$$X^{'} = \{x_{11}^{'}, x_{12}^{'}, ..., x_{1n}^{'} \quad x_{21}^{'}, x_{22}^{'}, ..., x_{2n}^{'} \quad ... \quad x_{m1}^{'}, x_{m2}^{'}, ..., x_{mn}^{'}\}$$

from cipher $S = \{s_1, s_2, ..., s_m\}$ abonent A analysis once $K^{Z} = \{k_1^{Z}, k_2^{Z}, ..., k_n^{Z}\}$ right to left, i.e. for each element of $S_j = \{s_1, s_2, ..., s_m\}, \quad j = 1,2,\ldots,m$

$$S_j = \begin{cases} S_j, & if\ \mathrm{S_j} < k_i^{Z} \\ S_j - k_i^{Z}, & if\ \mathrm{S_j} \geq k_i^{Z}, \quad i = 1,2,\ldots,n; \quad j = 1,2,\ldots,m \end{cases}$$

condition is checked. Here if condition $\mathrm{S_j} \geq k_i^{Z}$ is true (it means, on formation of the cipher $S_j = \{s_1, s_2, ..., s_m\}, \quad j = 1,2,\ldots,m$ - k_i^{Z} was used), we have to put ‘1’ in appropriate index of $X_j^{'}, \quad j = 1,2,\ldots,m$, alternatively we have to put ‘0’. Repeating this cycle for each element of $S_j = \{s_1, s_2, ..., s_m\}, \quad j = 1,2,\ldots,m$ we have modified open text, where length of them equals to n. Collecting on sequence all items of modified text $X_j^{'}, \quad j = 1,2,\ldots,m$ we have full present of open text – X. We divide it to parts where the length is the same as length of binary presentation of letter in Z alphabet. After putting them on appropriate succession we will restore open text $X = \{x_1, x_2, ..., x_l\}$ /11/.

The generating of box elements on the base of private parameter. For determination of number of box elements – n binary length of e^{Z} is taken, i.e. the minimum number n, which is response for condition $e^{Z} \leq 2^{n}, \quad n > 0$.

Box’s elements are superincreasing, i.e. the value of next element is greater than total sum of all previous. We can express such condition by following,

$$e_j > \sum_{i=1}^{j-1} e_i, \quad j = 2,3,\ldots,n.$$

Following instruction will be taken for generation of such vector. On the base on private parameter – e^Z pseudorandom sequence

$$T = \{t_1, t_2, \ldots, t_r\}$$

with the r –number of elements is generated and sorted in increasing order. The first element of classified vector is taken as the first box element, i.e. $k_1^Z = t_1$. For the next element – k_2^Z, we take such t_j, which responsible to

$$t_j > 2 \cdot k_1^Z, \quad j = 2,3,\ldots,r$$

Condition /11/.

Proof. It must be proved, that the clause $t_j > 2 \cdot k_1^Z, \quad j = 2,3,\ldots,r$ provides superincreasing characteristics, that is $k_{i+1}^Z > \sum_{j=1}^{j=i} k_j^Z$.

Using induction method, assume for the sequence

$$K^Z = \{k_1^Z, k_2^Z, \ldots, k_{i-1}^Z, k_i^Z, k_{i+1}^Z, \ldots, k_n^Z\}$$

is done the condition $k_i^Z > \sum_{j=1}^{j=i-1} k_j^Z$, then

$$k_{i+1}^Z > 2 \cdot k_i^Z \quad \Rightarrow \quad k_{i+1}^Z = k_i^Z + k_i^Z > k_i^Z + \sum_{j=1}^{j=i-1} k_j^Z = \sum_{j=1}^{j=i} k_j^Z \quad \Rightarrow \quad k_{i+1}^Z > \sum_{j=1}^{j=i} k_j^Z$$

from it feasibility of $k_{i+1}^Z > \sum_{j=1}^{j=i} k_j^Z$ is correct, which required to prove.

Thus, if the elements are taken from condition $k_{i+1}^Z > 2 \cdot k_i^Z$, they become superincreasing. Keeping on this way, instead of $k_i^Z \quad i = 1..n$ we should take t_j, for which the condition /11/

$$t_j > 2 \cdot k_i^Z, \quad j = 1,2,\ldots,r; \; i = 1,2,\ldots,n$$

is done, once we have superincreasing box $K^Z = \{k_1^Z, k_2^Z, \ldots, k_n^Z\}$ with the element n.

The example of crypto algorithm. Cryptosystem has two public parameters and, which are available for everybody and used for generation of shared private parameter with the Diffie-Hellman key generation algorithm. Latin alphabet is used for exchange message. Appropriate codes and binary present of letters are given below. In this example binary length of characters equals to 5, however in real applications it can be longer, for instance in web applications which use ASCII character encoding bit length equals to 8 and in international UNICODE encoding character bit length equals to 16.

Table 2.2. Bit length of character

a	b	C	d	e	f	g	h	i
1=00001	2=00010	3=00011	4=00100	5=00101	6=00110	7=00111	8=01000	9=01001
j	k	L	m	n	o	p	q	r
10=01010	11=01011	12=01100	13=01101	14=01110	15=01111	16=10000	17=10001	18=10010
s	t	U	v	w	x	y	z	
19=10011	20=10100	21=10101	22=10110	23=10111	24=11000	25=11001	26=11010	

Here in the Table 2.2, binary length $d = 5$.

1) Abonent A chooses $\alpha = 131$ and B $\beta = 37$ as private parameter, which used for generation of common private parameter $e_A^Z = e_B^Z = e^Z = 24$ via Diffie-Hellman algorithm. Presenting private parameter $e^Z = 24_{10} = 11000_2$ on binary view we have amount of box elements $n = 5$ and this is the minimum number which holds

$$e^Z \le 2^n \quad n > 0 \quad 26 \le 2^5 \quad \Leftrightarrow \quad 26 \le 32 \quad n > 0$$

clause.

For cipher message $X =$"*algoritm*" abonent B generates $r = 15$ pseudorandom number sequence

$$T = \{t_1, t_2, \ldots, t_r\} = \{17, 6, 4, 13, 9, 37, 20, 22, 49, 62, 43, 75, 93, 89, 95\}$$

on the base of private parameter $e^Z = 24$.

To obtain superincreasing box elements pseudorandom number sequence had to be sorted on increasing order

$$T' = \{t_1, t_2, \ldots, t_r\} = \{4, 6, 9, 13, 17, 20, 22, 37, 43, 49, 62, 75, 89, 93, 95\}$$

from here $k_1^Z = t_1 \;\Rightarrow\; k_1^Z = 4$. Ongoing such way and keep request

$$k_i^Z > 2 \cdot k_{i-1}^Z, \quad i = 2,3,\ldots,n$$

we can compute remained elements $K^Z = \{4, 9, 20, 43, 89\}$ base on pseudorandom number sequence $t_j, \quad j = 1,2,\ldots,r$.

Binary representation of message $X =$"*algoritm*"

$$a = 00001 \;\; l = 01100 \;\; g = 00111 \;\; o = 01111 \;\; r = 10010 \;\; i = 01001 \;\; t = 10100 \;\; m = 01101$$
$$X = "00001011000011101111100100100110100011101"$$

divided to $m = 8$ blocks, where each block has same length as binary presentation of letter in Z alphabet $n = 5$ and there we obtain modified open text. Here '0' is set on unfilled places. By scalar multiplication X' and K^Z we take integer cipher $S_j = \{s_1, s_2, \ldots, s_m\}, \quad j = 1,2,\ldots,m$.

$$X' = \{\; 0\;\; 0\;\; 0\;\; 0\;\; 1 \qquad 0\;\; 1\;\; 1\;\; 0\;\; 0 \qquad 0\;\; 0\;\; 1\;\; 1\;\; 1 \qquad \ldots \qquad 0\;\; 1\;\; 1\;\; 0\;\; 1\;\}$$
$$K^Z = \{4\;\; 9\;\; 20\;\; 43\;\; 89 \qquad 4\;\; 9\;\; 20\;\; 43\;\; 89 \qquad 4\;\; 9\;\; 20\;\; 43\;\; 89 \qquad \ldots \qquad 4\;\; 9\;\; 20\;\; 43\;\; 89\}$$
$$S = \{4 \cdot 0 + \ldots + 89 \cdot 1 = 89 \qquad 4 \cdot 0 + \ldots + 89 \cdot 0 = 29 \qquad 4 \cdot 0 + \ldots + 89 \cdot 1 = 152 \ldots \; 4 \cdot 0 + \ldots + 89 \cdot 1 = 118\}$$

Cipher $S = \{89, 29, 152, 161, 47, 98, 24, 118\}$ is send to abonent A /11/.

2) Maintaining foregoing scheme abonent A also generates $n = 5$ and $K^Z = \{4, 9, 20, 43, 89\}$. For getting modified open text

$$X' = \{\; x'_{11}, x'_{12}, \ldots, x'_{1n} \quad x'_{21}, x'_{22}, \ldots, x'_{2n} \quad \ldots \quad x'_{m1}, x'_{m2}, \ldots, x'_{mn} \;\}$$

from cipher $S = \{89, 29, 152, 161, 47, 98, 24, 118\}$ abonent A analysis once

$$K^Z = \{k_1^Z, k_2^Z, \ldots, k_n^Z\}$$

right to left, i.e. for each element of $S_j = \{s_1, s_2, \ldots, s_m\}, \quad j = 1,2,\ldots,m$. In case of the last element $s_8 = 118$, at first $x'_1 =$"" and after

$s_8 \geq k_5^Z (118 \geq 89) \quad x_1^{'} = "1" \quad s_8 = 118 - 89 = 29;$

$s_8 < k_4^Z (29 < 43) \quad x_1^{'} = "01" \quad s_8 = 29;$

$s_8 \geq k_3^Z (29 \geq 20) \quad x_1^{'} = "101" \quad s_8 = 29 - 20 = 9;$

$s_8 \geq k_2^Z (9 \geq 9) \quad x_1^{'} = "1101" \quad s_8 = 9 - 9 = 0;$

$s_8 < k_1^Z (0 < 4) \quad x_1^{'} = "01101" \quad s_8 = 0;$

calculations we have

$$X^{'} = "00001 \quad 01100 \quad 00111 \quad 01111 \quad 10010 \quad 01001 \quad 10100 \quad 01101"$$

modified open text. Dividing it to parts, which the length is the same as length of binary presentation of letter in Z alphabet $d = 5$,

$$00001 = a \;\; 01100 = l \;\; 00111 = g \;\; 01111 = o \;\; 10010 = r \;\; 01001 = i \;\; 10100 = t \;\; 01101 = m$$

and set them on appropriate sequence we restore open text $X =$ "*algoritm*" /11/.

While this crypto system is hybrid crypto system it includes asymmetric key exchange generating Diffie-Hellman algorithm and symmetric encryption algorithm for further information exchange, using previously generated key as a seed parameter of actual encryption keys. As robustness and length of cipher depends on key length, different key length can be used, according to security of information. For instance, in multi level management systems such as government administering, different key bit length can be used depending on priority of authority or in Bank transfer system key bit length should be used according to transfer amount.

Conclusion

To provide rich services available in e-document exchange systems it has to be extensible and optimal for manipulation and storage. Structure of e-document has to be extensible in multi level government management system, where in this purpose nowadays' wide technology is XML format. As XML format is portable to transfer through network, it can be used to construct complex structure of e-documents. Rich structure of XML document is constructing by W3C

organization, while organization is intensively working to provide encryption, digital sign inside same XML document. In this purpose, XML Signature and XML Encryption was developed and proposed.

Fully implementation if e-document exchange can be gained if core application is able to support those options, while in our case core technology is database management system. To provide confidential information flow in e-document exchange system, we developed 10 database tables, while each of them has appropriate operational functionality. Beside database architecture exact steps of e-document flow is defined, taking in account all cases.

As we said before, confidential information exchange can be achieved applying cryptographic tools, whilst in this project we used four major cryptographic tools – encryption, hash function, digital signature and data compression.

Author of final qualifying work, learned cryptography during plenty of time, and we built new encryption algorithm, which it can be applied in e-document exchange also.

III. Web site applied cryptographic tools on working with confidential information in electronic-document exchange

Software's which enables electronic document exchange, usually calls as *Document Management Systems*. There are a number of solutions such as Alfresco, which is widely used open source document management system, Content Repository API for Java ™ Technology (Day software). These projects are supposed to manage electronic documents, and as my work devoted to security options of electronic document exchange systems, here I developed web based electronic document exchange solution, where different cryptographic tools were used to accomplish secure document exchange.

Chapter 3.1 is devoted to discussion of used technologies, we development language, RDBMS (relational database management system) and web environment (server). Besides this, explanation of cryptographic tools and libraries, such as encryption, electronic sign generation and verification algorithms, which were used to accomplish security of e-document exchange, are carried out.

Chapter 3.2 illustrates working principals and steps of electronic document exchange system, while each steps of user is illustrated and explained in more detail.

Chapter 3.3 continues discussion of Chapter 3.1 comparing algorithms, by different criteria's, such as, used operation system resources, time, and robustness of e-sign and encryption algorithms.

3.1. Designing electronic-document exchange technologies, environment and cryptographical libraries

Although web server programs differ in detail, they all share some basic common features /34/.

1. **HTTP**: every web server program operates by accepting HTTP requests from the client, and providing an HTTP response to the client. The HTTP response usually consists of an HTML document, but can also be a raw file, an image, or some other type of document (defined by MIME-types). If some error is found in client request or while trying to serve it, a web server has to send an error response which may include some custom HTML or text messages to better explain the problem to end users.

2. **Logging**: usually web servers have also the capability of logging some detailed information, about client requests and server responses, to log files; this allows the webmaster to collect statistics by running log analyzers on log files.

In practice many web servers implement the following features also:

1. **Authentication**, optional authorization request (request of user name and password) before allowing access to some or all kind of resources.

2. **Handling of static content** (file content recorded in server's filesystem(s)) and dynamic content by supporting one or more related interfaces

3. **HTTPS support** (by SSL or TLS) to allow secure (encrypted) connections to the server on the standard port 443 instead of usual port 80.

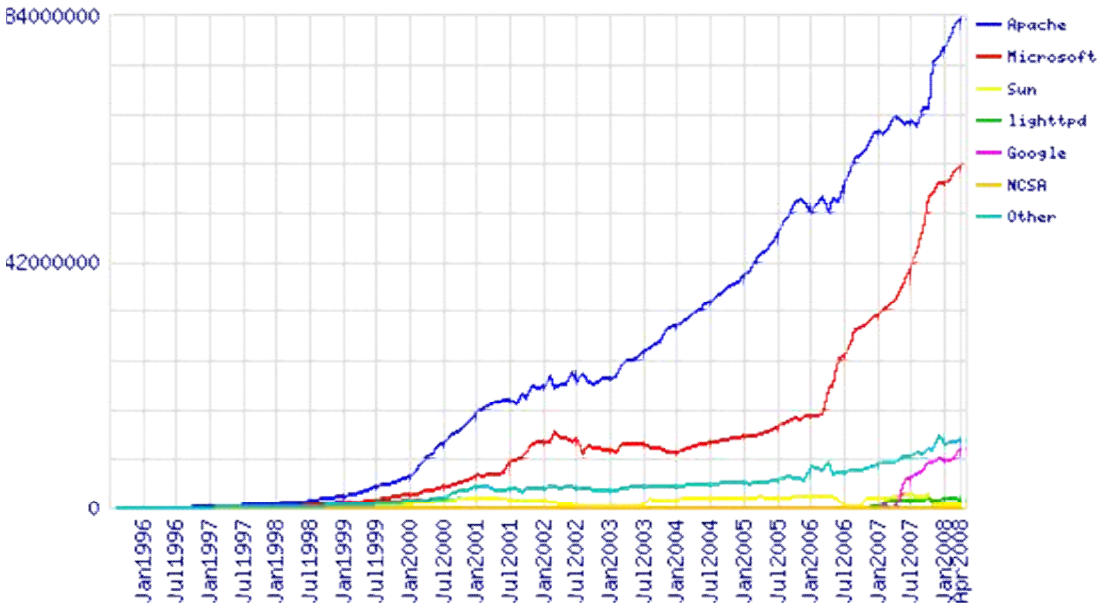
4. **Content compression** (i.e. by gzip encoding) to reduce the size of the responses (to lower bandwidth usage, etc.).

5. **Virtual hosting** to serve many web sites using one IP address.

6. **Large file support** to be able to serve files whose size is greater than 2 GB on 32 bit OS.

7. **Bandwidth throttling** to limit the speed of responses in order to not saturate the network and to be able to serve more clients.

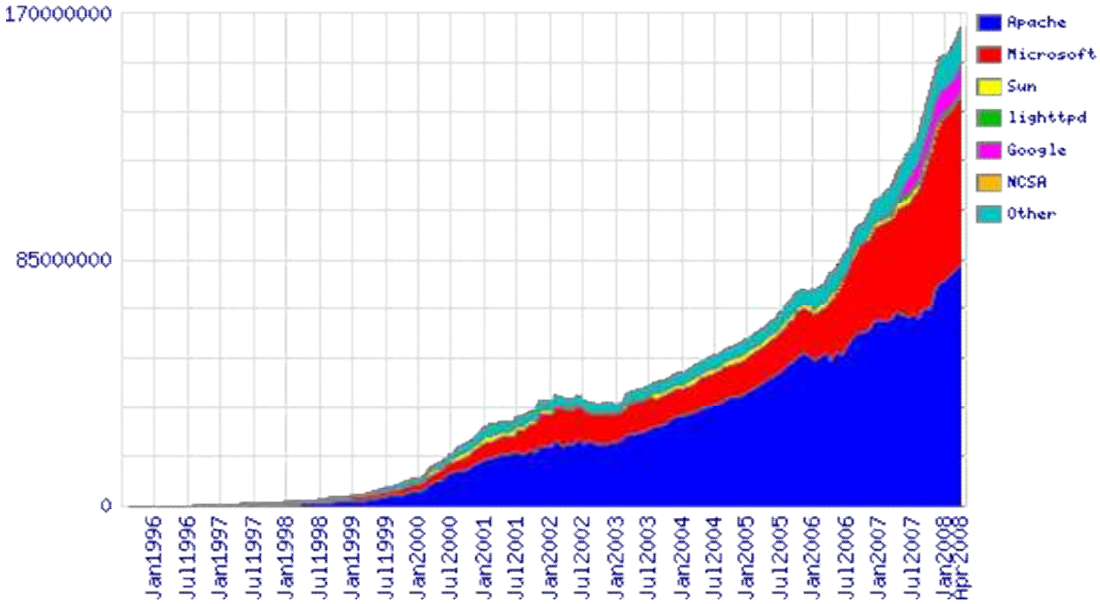
Apache Tomcat is a *Servlet* container developed at the **Apache Software Foundation** (ASF). Tomcat implements the *Java Servlet* and the *JavaServer Pages (JSP)* specifications from Sun Microsystems, and provides a "pure Java" HTTP web server environment for Java code to run /34/.



Source: http://www.netcraft.com/

Figure 3.1. Totals for top servers across all domains

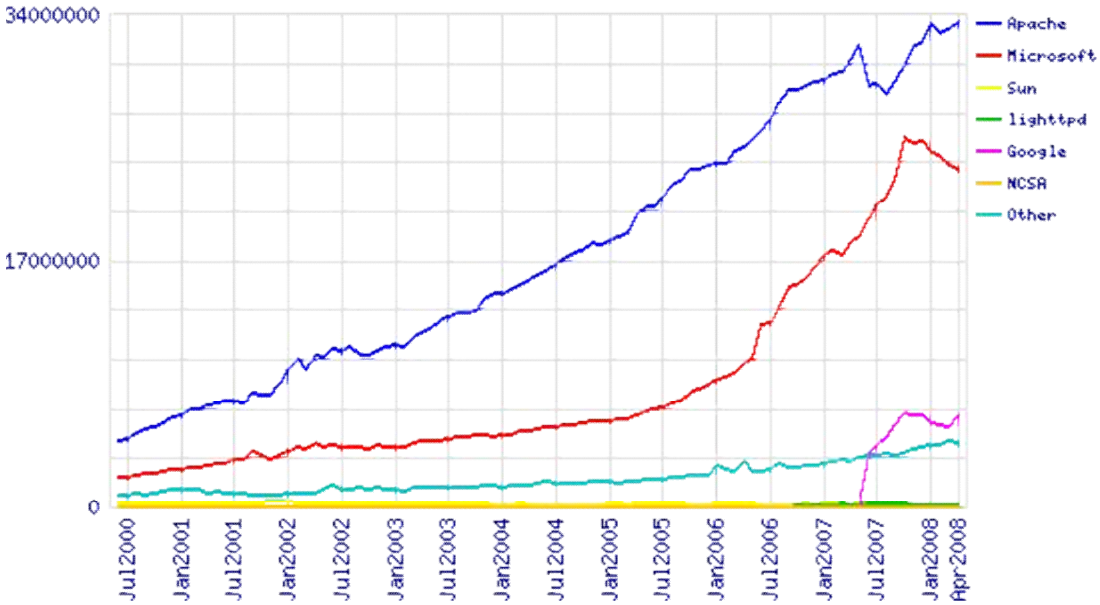
As we can see in Figure 3.1 there are number of web servers across the web, but Apache Server is most widely used because, it has best suitable features such as, Open Source Software to some extend and it provides security issues. Although Apache supports a number of web development languages, but Java Server Pages (JSP) and Servlet web development is one of the most widely used server technology, after PHP scripting language. The differences are, while PHP is lightweight scripting language, it can not be used for complex business application and via PHP web developer can't offer strong security as JSP. As Java Server Pages (JSP) and Servlet technologies are based on pure Java, Object Oriented Programming language, they can offer robust security and more other opportunities to web developers.



Source: http://www.netcraft.com/

Figure 3.2. Change in server numbers across all domains

As shown in Figure 3.2 Apache server start its development from early 1998th and it is dramatically increasing in number year-to-year and although Microsoft's web server growth (IIS) can be comparable with it's, but Apache's growth accounts twice more.



Source: http://www.netcraft.com/

Figure 3.3. Totals for active servers across all domains

Figure 3.3 shows currently active web servers across worldwide domains, however as illustrated in graph, Apache server is even growing and widely using

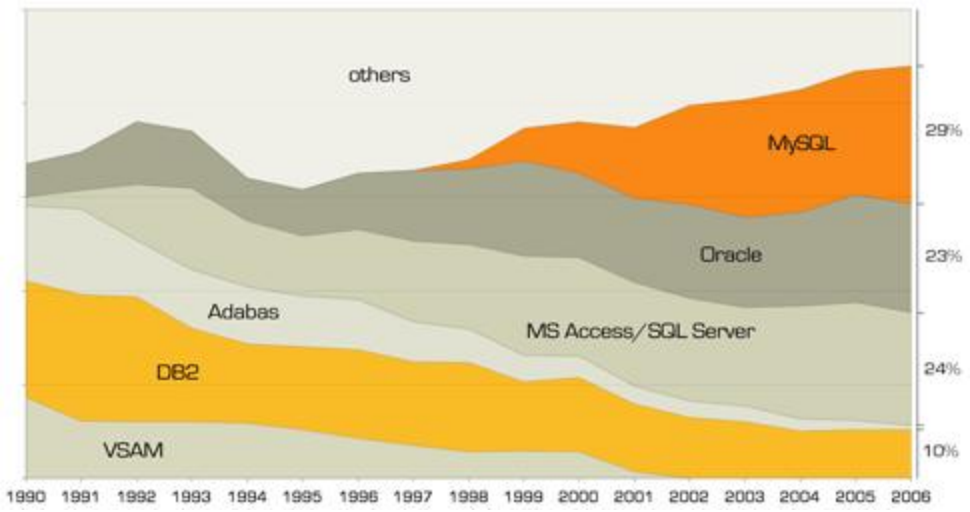
web server, while even in 2000 it encouraged more then 500 000 domains and now it comes to near 34 million domains which use Apache web server.

Implementation and market share of MySQL database management server. MySQL is the world's most popular open source database. With over 50,000 downloads per day, MySQL continues to be the choice for a broad range of database developers, DBA(Database Administrator)s and IT managers who want a high performance database that is reliable, affordable, and easy to use /33/.

MySQL has gained 25% market share in overall database usage by developers in the last two years, according to data drawn from a range of recent multi-client surveys published by Evans Data Corporation, a vendor-neutral third-party market research firm. The data shows 40% database usage by developers, up from 32% two years ago. Evans Data Corp. noted that with more and more developers using Open Source (65% in North America in Fall 2006), usage of MySQL is projected to continue to increase in the future.

According to the recent JoinVision study "Open Source in the Fast Lane", IT specialists indicated they deploy MySQL 30% more frequently than Oracle, SQL Server or DB2 Figure 3.4.

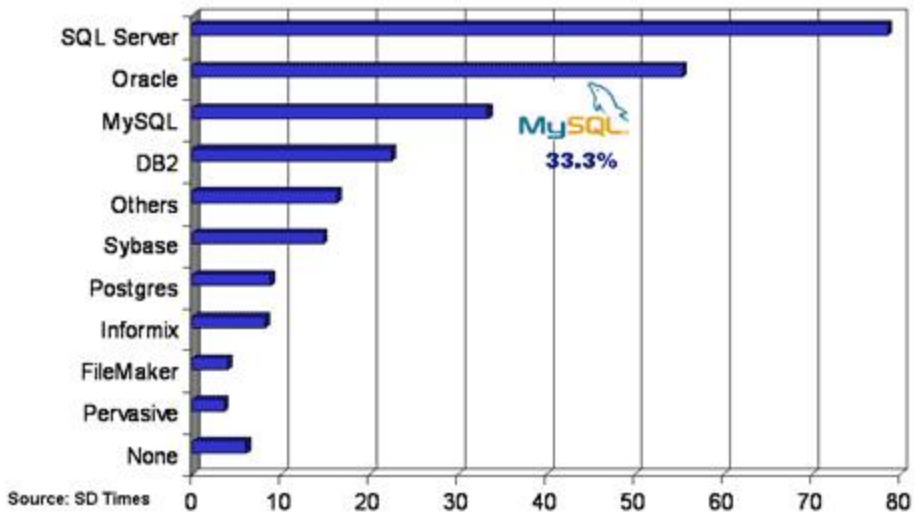
Furthermore, this study concludes that open source is a key component of today's IT infrastructure, and that the market share of open source technology is increasing.



Source: JoinVision E-Services GmbH, July 2006

Figure 3.4. IT experts today most often use MySQL

In an article "Relational Databases Rule the Roost" published in SD Times in July 2004, MySQL was identified as the number 3 "Top Deployed Database" in a survey of 934 readers Figure 3.5. MySQL was more broadly deployed than DB2, Informix, Postgres or Sybase. Respondents cited familiarity with the database, reputation of the database vendor and lowest development costs as the top 3 factors that led to their database choice.



Source: http://www.mysql.com/why-mysql/marketshare/

Figure 3.5. MySQL is the #3 database among those polled

As we illustrated conclusion of different vendor-neutral third-party market research firms, DBMS MySQL and Apache web server is most suitable for large scale projects. As soon as both of them are open source, their development will be carried out by more software companies. As I going to implement my e-document exchange system to large scale companies and government agencies, MySQL and Apache Server with JSP and Servlet support is most suitable development tool for me, hence I choose this technologies.

Developed cryptographic libraries. To support secure e-document exchange we developed some cryptographic libraries which was used to secret key agreement, encryption/decryption, hashing and etc. As Java Object Oriented Programming (OOP) language used to make project and libraries, we create *.java files which after compile become a *.class file and these *.class files serve us as working libraries. Here we'll show main classes, while each of them has own purpose.

In Java language specification classes which do some logic together has to be compiled inside one package. For cryptographic libraries I made *nodir.crypto* and *nodir.java.util* packages, while there are number of classes, which realize main cryptographic algorithms shown in Table 3.1.

Table 3.1.Main cryptographic classes

Name	Package	Summary of class
SHA_256	nodir.crypto	Used to evaluating hash value (digest) of string, where length of digest is 256 bits. An algorithm realizes all steps proposed in FIPS PUB 180-3 specification.
KattaSon	nodir.crypto	Used to do calculation of big numbers, which are from 200 to 800 digits in length.
KattaKasrSon	nodir.crypto	Manipulates decimal number encapsulated by bytes. Has method for addition, remain, modular division and etc. works same as core java.math.BigDecimal class.
KattaButunSon	nodir.crypto	Does calculation of big integers and has all basic mathematic calculations, including cryptographic methods. Actually, private analog of java.math.BigInteger class. Object encapsulated as array of bytes.
DiffieHellman	nodir.crypto	Used to generate secret key pair, defined in Diffie-Hellman key agreement cryptosystem.
DESEncryption	nodir.crypto	Does encryption and decryption of message, returning byte representation of cipher. Method realizes steps defined in FIPS PUB 46 “Data Encryption Standard” specification. As development tool I used third party library provided from Bouncycastle free library.
Arrays	nodir.java.util	Manipulates arrays, which elements are *KattaButunSon* and *KattaKasrSon*, sorting them as required in *YSAUmumiy* algorithm. Method uses fast binary sort algorithm to rank elements in ascending order.
YSAUmumiy	nodir.crypto	Class where I realized my own cryptographic algorithm called Private Box Algorithm. Used to encryption/decryption of messages, defined in Chapter 2.4.

Next we will explain some main classes shown in the Table 3.1.

SHA_256. Used to evaluating digest of message, where length of digest is 256 bits. Cryptographic algorithm has implementation for longer messages, returning 512 and 1024 bit length digest values, defined in FIPS PUB 180-3. Description of method given in the Table 3.2.

Table 3.2 SHA_256 class's method summary

Return type	Method description
`String`	**`getHash`**`(String text)` Returns digest value of message given as `text` argument.
`String`	**`doHash`**`(String paddedString)` Encapsulated `private` method, does message digest after padding.
`String`	**`lastCorrect`**`(String must8)` Pads given argument to 8 bit length, which requires to calculate hash value.
`String`	**`padding`**`(String unpadded)` Pads given `unpadded` string to `unpadded.length` mod 512 = 448.
`String[]`	**`toNormalBinaryString`**`(String text)` Returns `binary representation of each character of given text with given encoding in String array element. In our case encoding ASCII was used.`

YSAUmumiy. Table 3.3 shows methods of YSAUmumiy class, where this class used to encrypt/decrypt messages, it uses special class called `KattaButunSon` to manage all operation. For instance, all box elements are stored in class member called elements, which encapsulated with **`getElements`** method. Consequently other method as **`deshifr`**, **`shifr`** does appropriate tasks of encryption.

Table 3.3 YSAUmumiy class's method summary

Return type	Method description
`KattaButunSon[]`	**`getElements`**`(KattaButunSon key)` Returns elements of Box from given `key` using it as a seed.
`KattaButunSon[]`	**`shifr`**`(String openText, KattaButunSon key)` Returns cipher of given `openText` using given `key`.
`String[]`	**`toNormalBinaryString`**`(String text)` Returns `binary representation of each character of given text with given encoding in String array element. In our case encoding ASCII was used.`
`String`	**`deshifr`**`(KattaButunSon[] shifrText, KattaButunSon key)` Returns plaintext cipher given as `shifrText` using given `key`.

KattaButunSon. This class is responsible to manipulate big integer numbers, doing multiplition, addition, remaining and modular divide of one

KattaButunSon to the other. Table 3.4 describes constructors, which uses to construct objects and methods which allows different arithmetic calculations on KattaButunSon object.

Table 3.4 KattaButunSon class's description

Constructor summary	
KattaButunSon(int number) Constructs **KattaButunSon** object which has value number	
KattaButunSon(long number) Constructs **KattaButunSon** object which has value number	
KattaButunSon(String number) Constructs **KattaButunSon** object from given string representation, which has value number	
KattaButunSon(byte[] number) Constructs **KattaButunSon** object which has value number stored in array of bytes	
Method summary	
Return type	Method description
byte[]	**getBytes**() Returns value of KattaButunSon in array of bytes.
KattaButunSon	**valueOf**(String strValue) Returns KattaButunSon object constructed from string strValue.
String	**toString**() Returns String representation of KattaButunSon object.
boolean	**equals**(KattaButunSon kbs) Used to check equality of this and kbs objects. Returns true if two objects are equal, false in other case.
KattaButunSon	**remainder**(KattaButunSon kbs) Returns remained value of kbs divided to this object.
KattaButunSon	**remain**(KattaButunSon kbs) Returns remain value of (this-kbs) as KattaButunSon object.
int	**compareTo**(KattaButunSon kbs) Returns: -1, 0 or 1 as this is numerically less than, equal to, or greater than kbs subsequently.
KattaButunSon	**add**(KattaButunSon kbs) Returns addition value of (kbs+this) as KattaButunSon object.
KattaButunSon	**multiply**(KattaButunSon kbs) Returns multiplying value of (this*kbs) as KattaButunSon object.
String	**toBinaryString**() Returns binary representation of this object.

Here we describe main classes of developed cryptographic library, while other classes are supplementary classes and used to properly illustrated classes. For instance, `KattaKasrSon` which exists in the same package (nodir.crypto) as `KattaButunSon` has similar methods described in the Table 3.4 except second class manipulates decimal numbers, used in `YSAUmumiy` class.

3.2. Planning confidential document flow in electronic-document exchange system

Web site accomplishes to one of the most important task of electronic commerce, which is electronic document exchange system and provide processing with confidential information in it. Here we use commonly available and special cryptographic tools which are applied to provide confidential information exchange system. Here we use four types of cryptographic tools, which are e-sign, encryption, hash function and data compress technique to enhance bandwidth between users and central registration office. While working algorithm is explained in Chapter 2.2 in more detail let's look to actual work within real taken pictures.

As shown in Figure 3.6. first user enters to welcome page, where mission and vision of this project is explained, here user can learn about project.

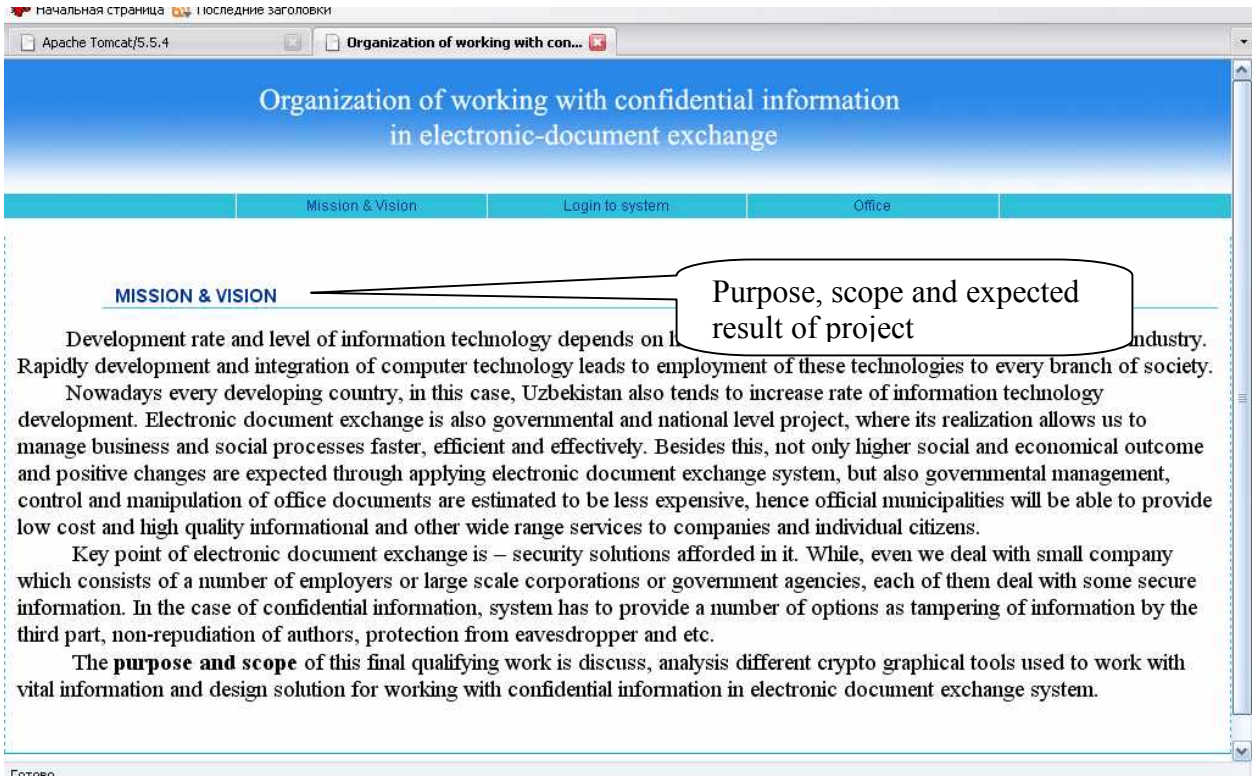


Figure 3.6. Welcome page of confidential e-document exchange system

Users who have pre-registered account are able to use system to exchange confidential e-documents Figure 3.7. For that user clicks *Login to System*, he will be asked for a login and password, if he had been registered before, if not, user can easily sign up choosing *Sign up* option.

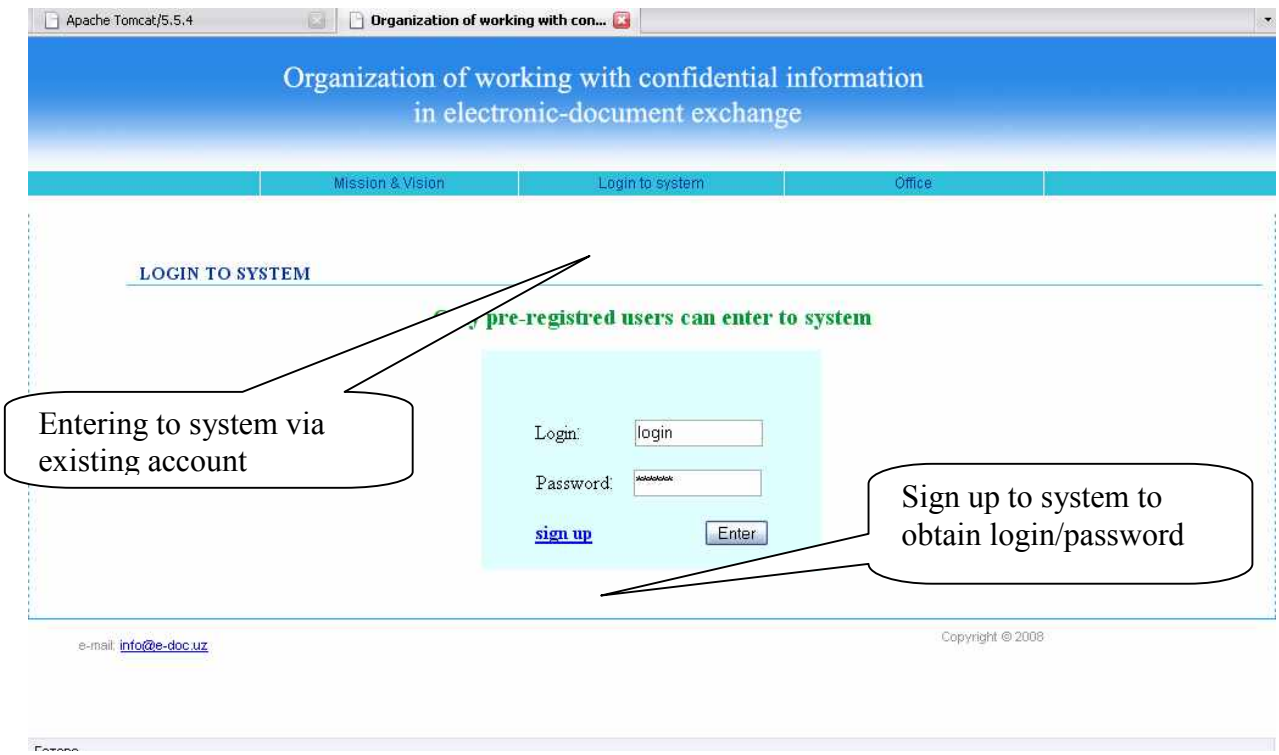


Figure 3.7. Enter to system with pre-registered account

Choosing *sign up* option user comes to registration page Figure 3.8 and he is required to present exact information about him/herself filling all necessary fields, while here provided information is used for further document exchange. At the registration page user will be asked for first name, last name, surname, secondary e-mail, phone, physical address and at the same time user has to choose login and password for further usage of system. System can register only one user with login and if user chooses previously registered login, he will be prompted to select another login. As user completes account registration, he will be able to log in via previously given login and password and user can exchange electronic documents only with other users of same system.

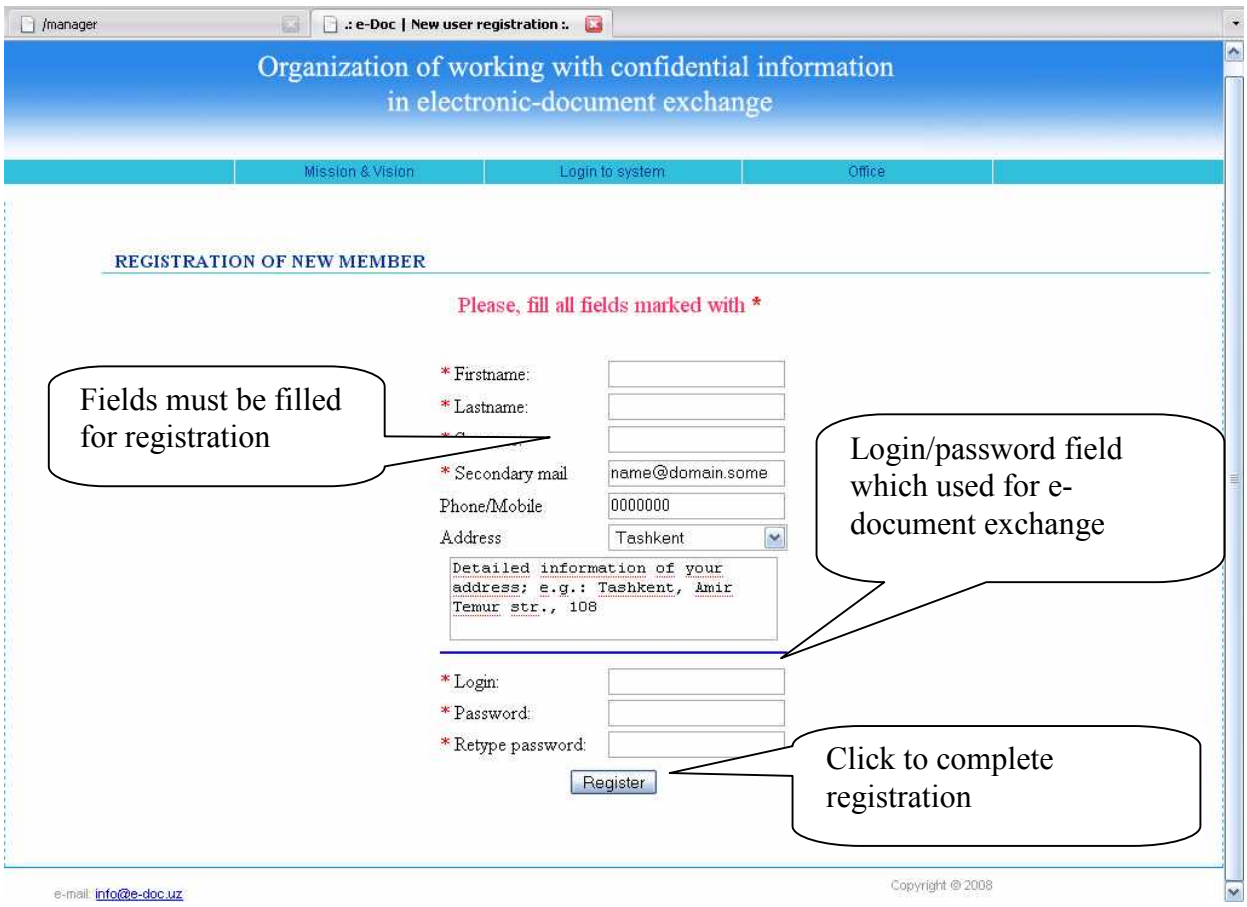


Figure 3.8. New user registration

Creating account in system user becomes a member and he/she can fully use system functionality. Here user can compose e-mail attaching e-document, see incoming messages and e-documents to it, check sent message's status (whether it

registered in office and delivered to destination or not yet) and look overall statistics of exchanged e-documents, look Figure 3.9.

Compose option of system allows users to compose messages and attach electronic documents to his/her mail, signing it with either with private or symmetric secret keys. As in a real document management system, composed mail will not be delivered to destination without registration of central office.

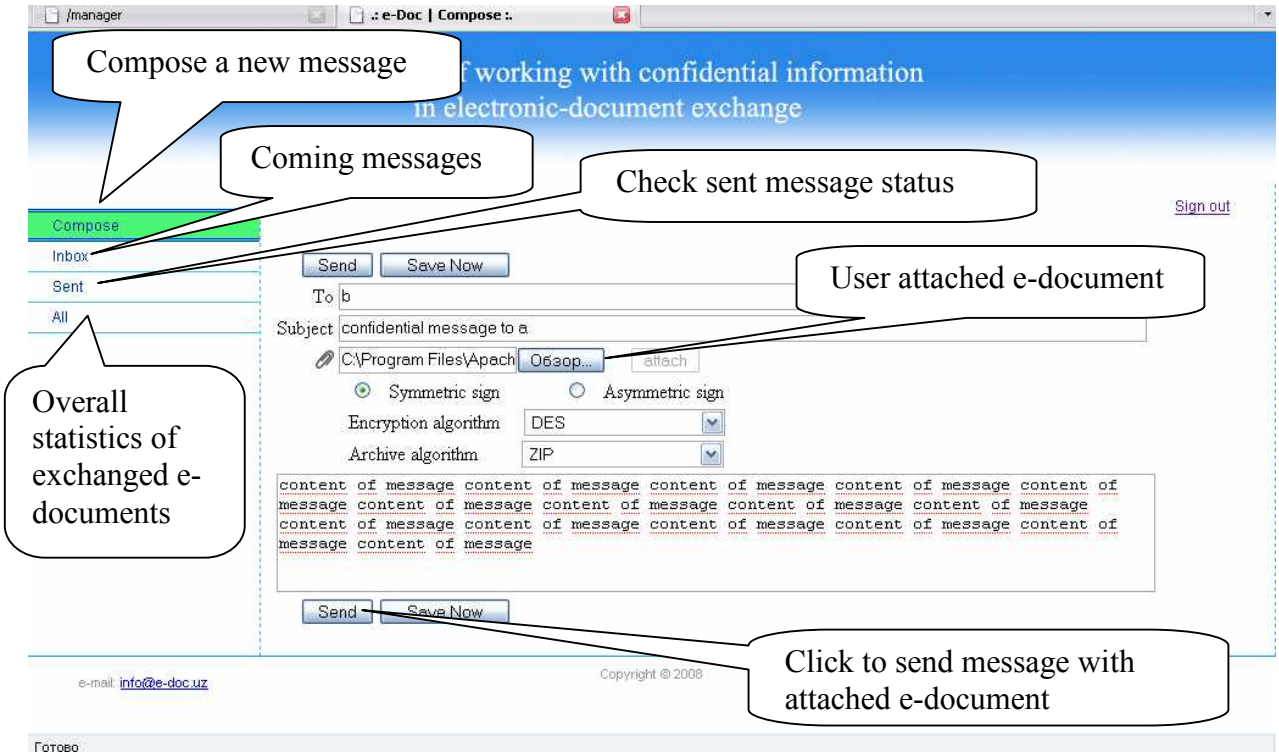


Figure 3.9. Compose message attaching e-document

When user clicks *Send* button user sees properties of sent message and it will be automatically delivered to central registration office, where *administrator* will check properties of sent message.

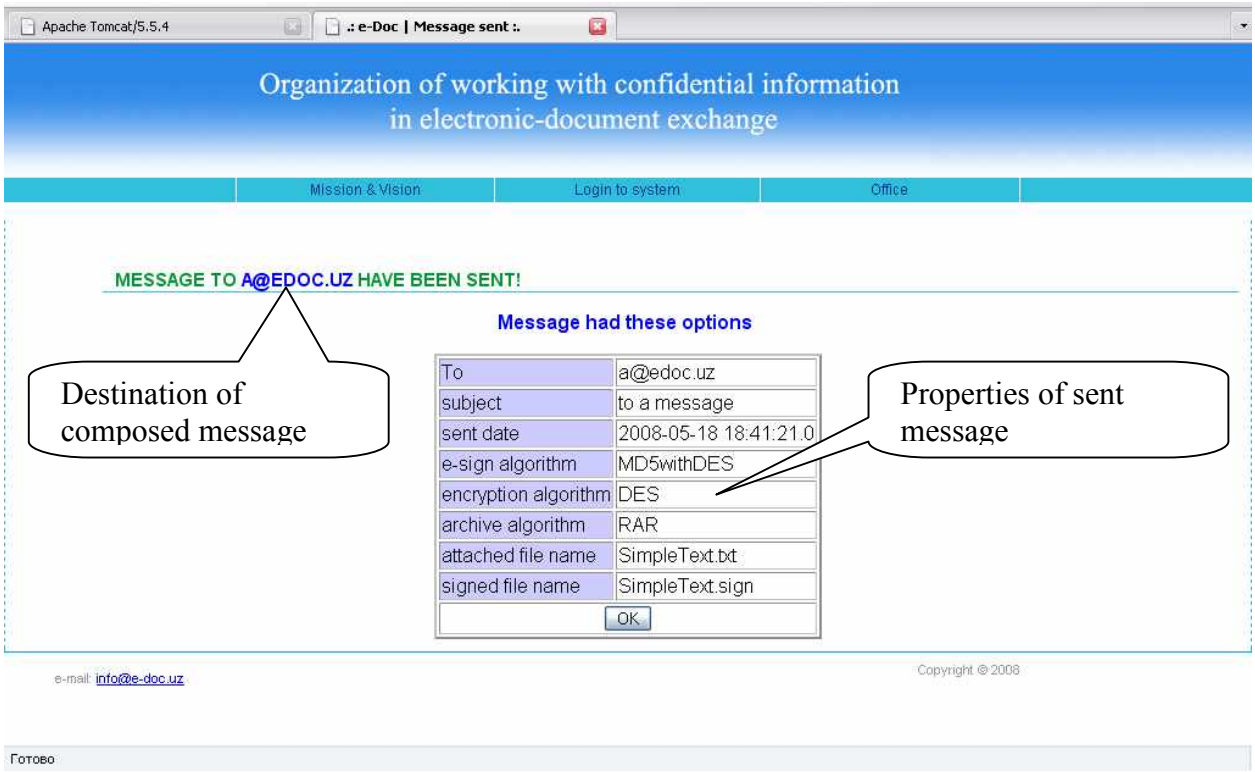


Figure 3.10. Complete properties of sent message

Choosing *Sent* option check sent message's status, whether it registered in office and delivered to destination or not yet. Here will be presented registration date and registration ID of message if message was registered by central registration office.

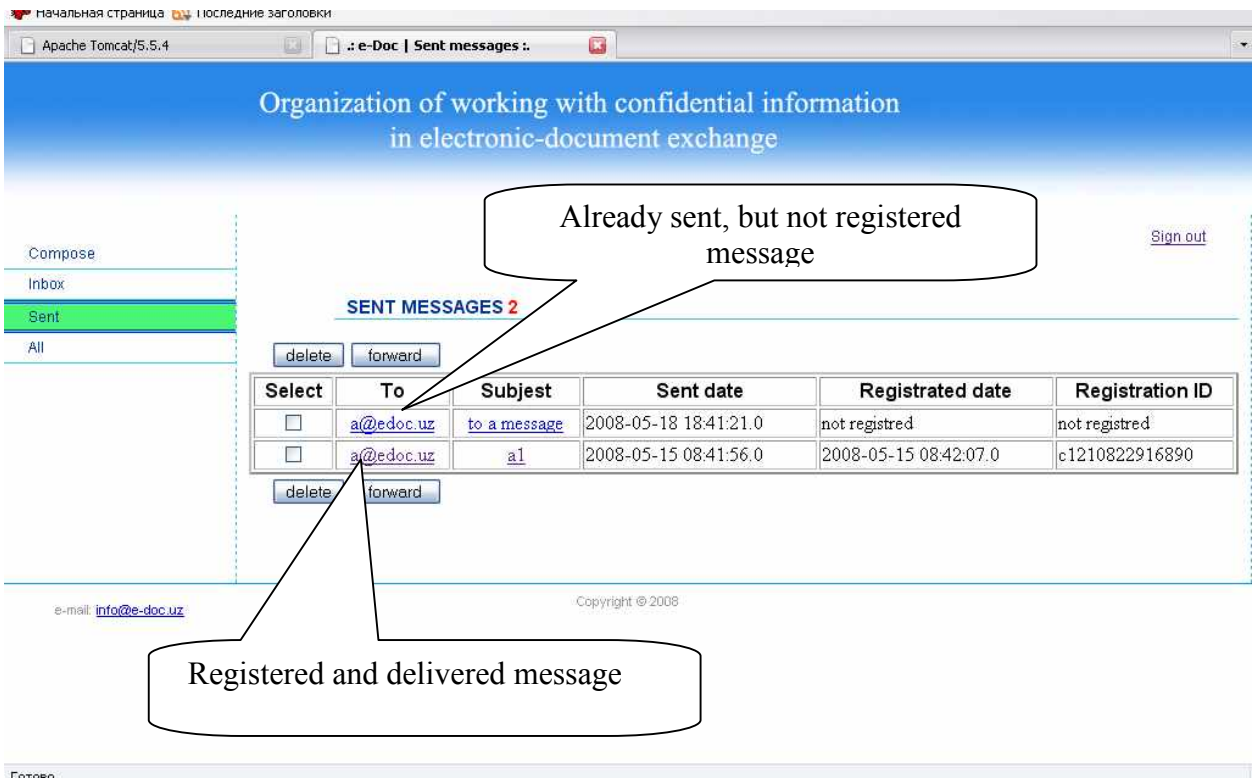


Figure 3.11. Sent message of user

Every time besides seeing overall message status, user is able to see detailed properties of sent message clicking *Subject* or receiver of the message Figure 3.12.

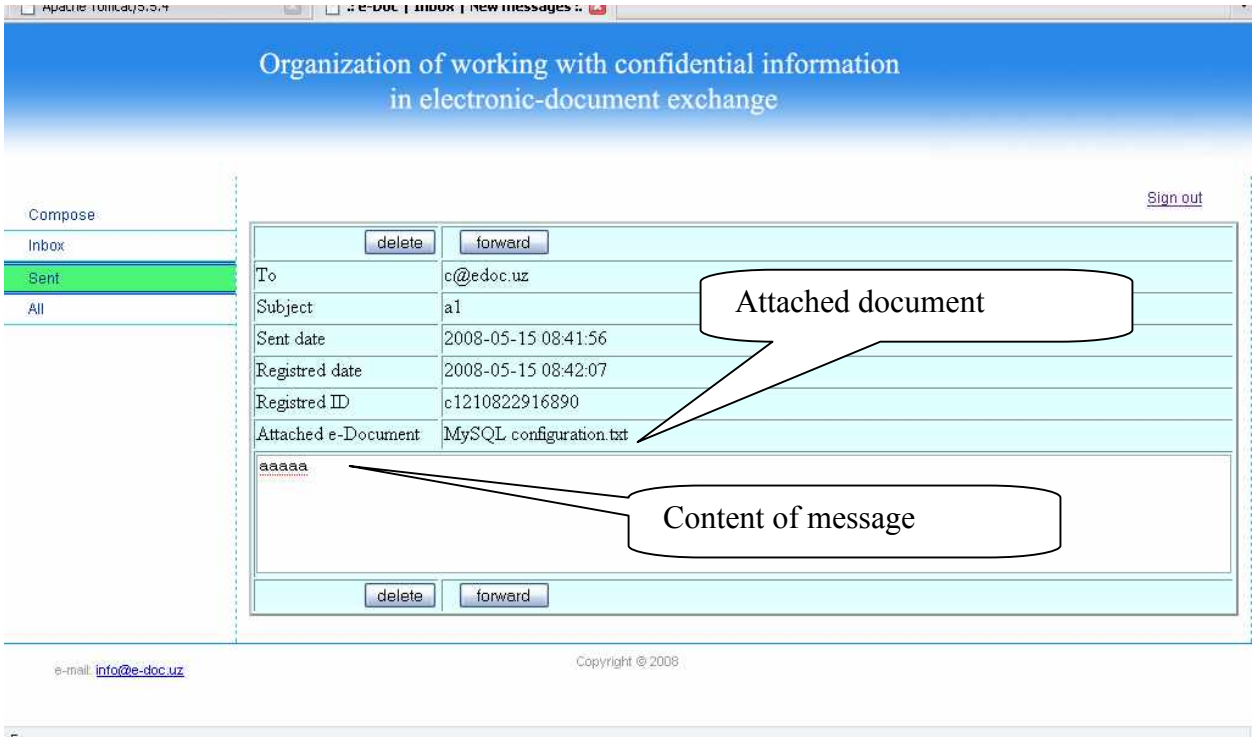


Figure 3.12. Sent message detail

Sent message automatically comes to office and office manager will be prompted to registration of new message with unique ID Figure 3.13, where it immobilizes users from repudiation. Only pre-registered administrators are able to *login* as managers and this authentication is accomplished with appropriate login/password.

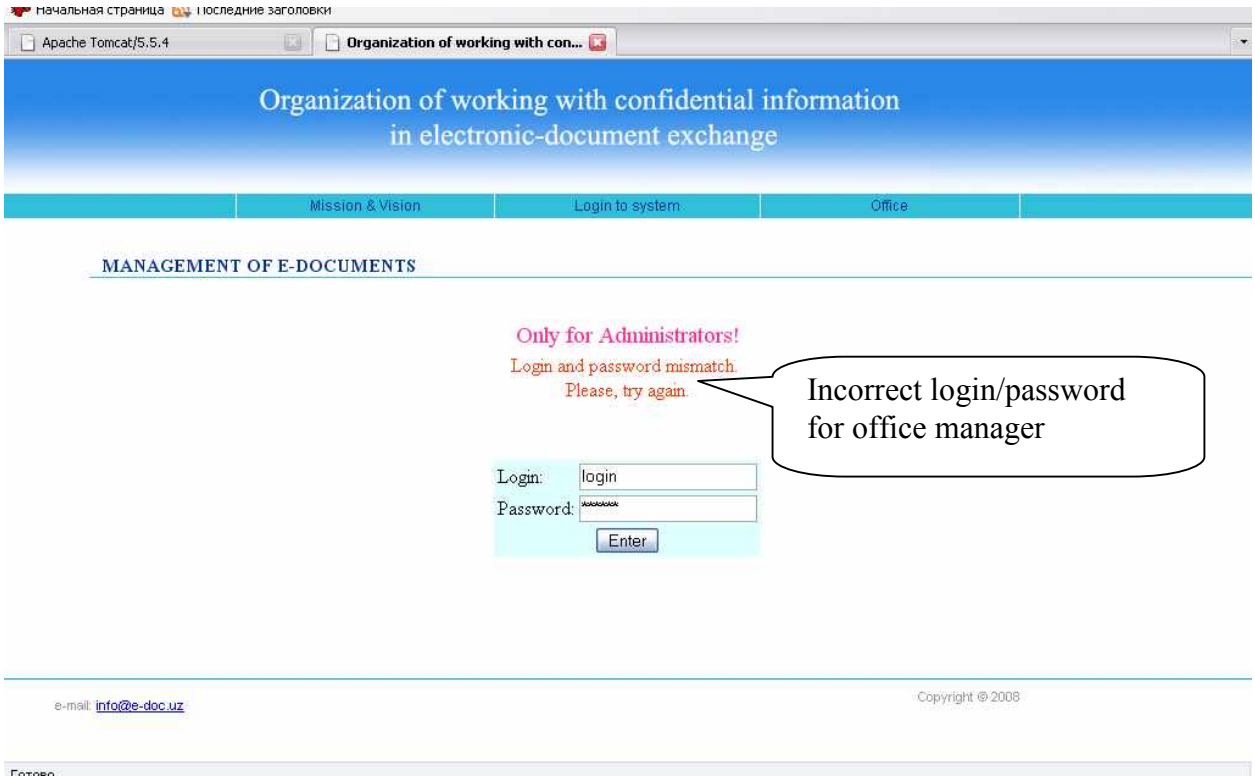


Figure 3.13. Login page of registration office manager

Office manager checks only general options of message, like contains appropriate subject, author and etc. However, office manager has not any access to the content of the message or to the attached electronic document, while manager only checks overall properties of the message Figure 3.14. Where his/her duty is only register incoming message, document and deliver it to destination.

Registration is done by selecting appropriate messages and clicking *Send* button and every time count of new registration messages are shown after login of manager, while in this case login of manager is *admin* as shown in Figure 3.14.

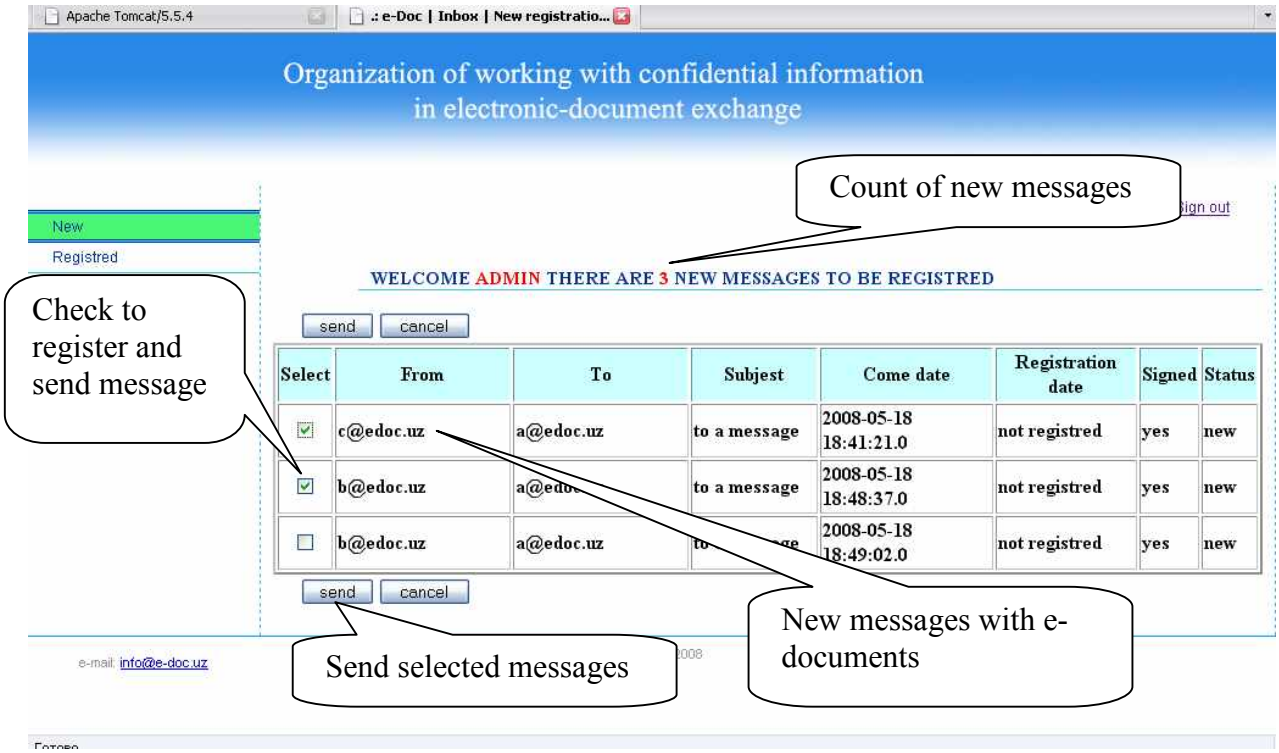


Figure 3.14. Registration of messages by office manager

Accordingly *admin* has list of pre-registered messages with overall information which includes *subject, registration ID, registration date, sent date* as shown in Figure 3.15.

Figure 3.15. Previously registered messages with e-documents

After successfully registration of e-documents in office, actual recipient of user will receive message with attached e-document, where *registration ID* and

sent date is shown in table, where registration date is not date of recipient but actual compose date of actual sender. Anytime user can take detail information about used encryption, compression and digital sign algorithms.

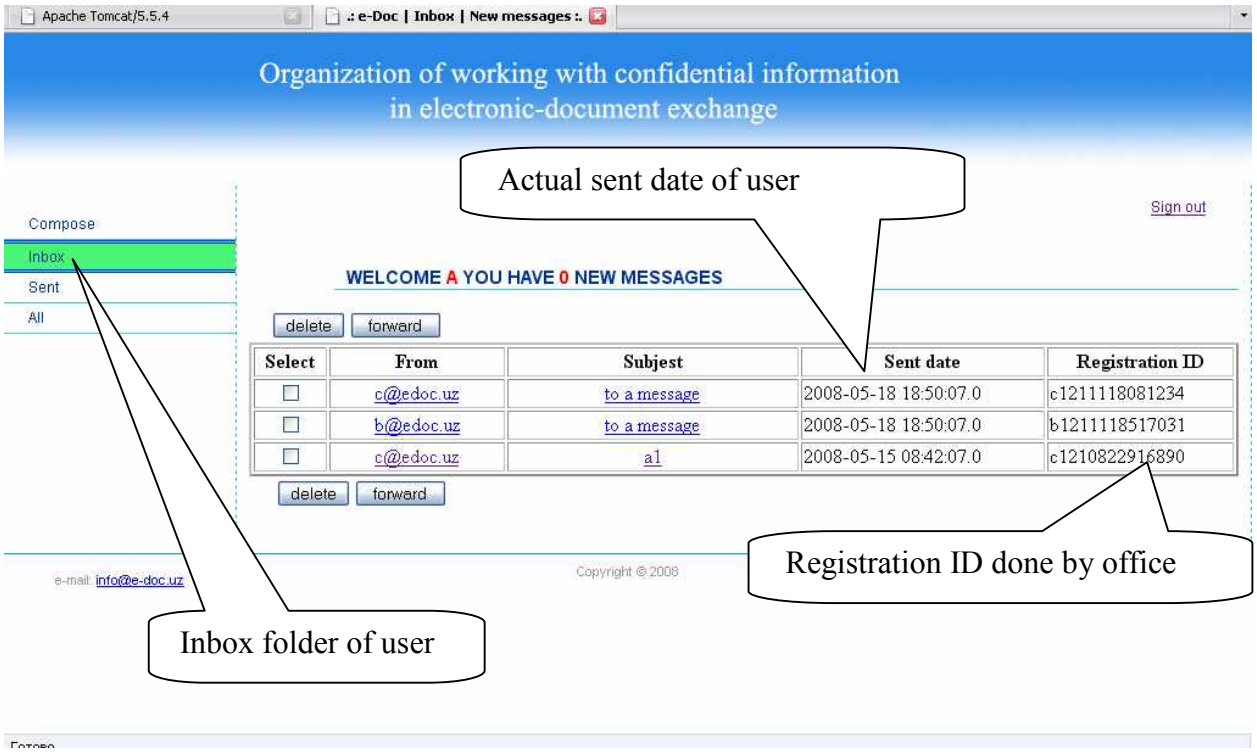


Figure 3.16 Registered e-document list of recipient user

Clicking in the subject or in author name user is able to see content of message, where all detail information is described in message and clicking name of attached e-document user is able to check validity of digital sign Figure 3.17, while after user's click system automatically load a secret key and check validity by given algorithm.

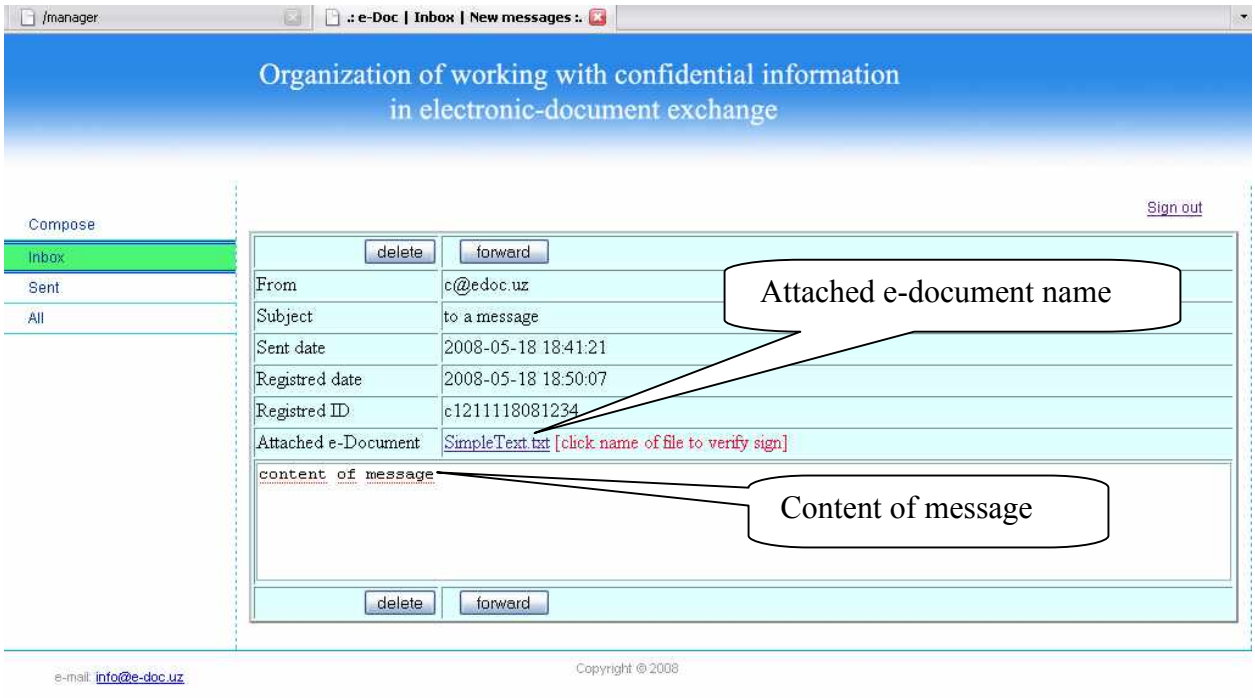


Figure 3.17. Registered e-document content

As shown in the Figure 3.18, if validation is complete and successful, user will be given the content of e-document associated with message registration ID.

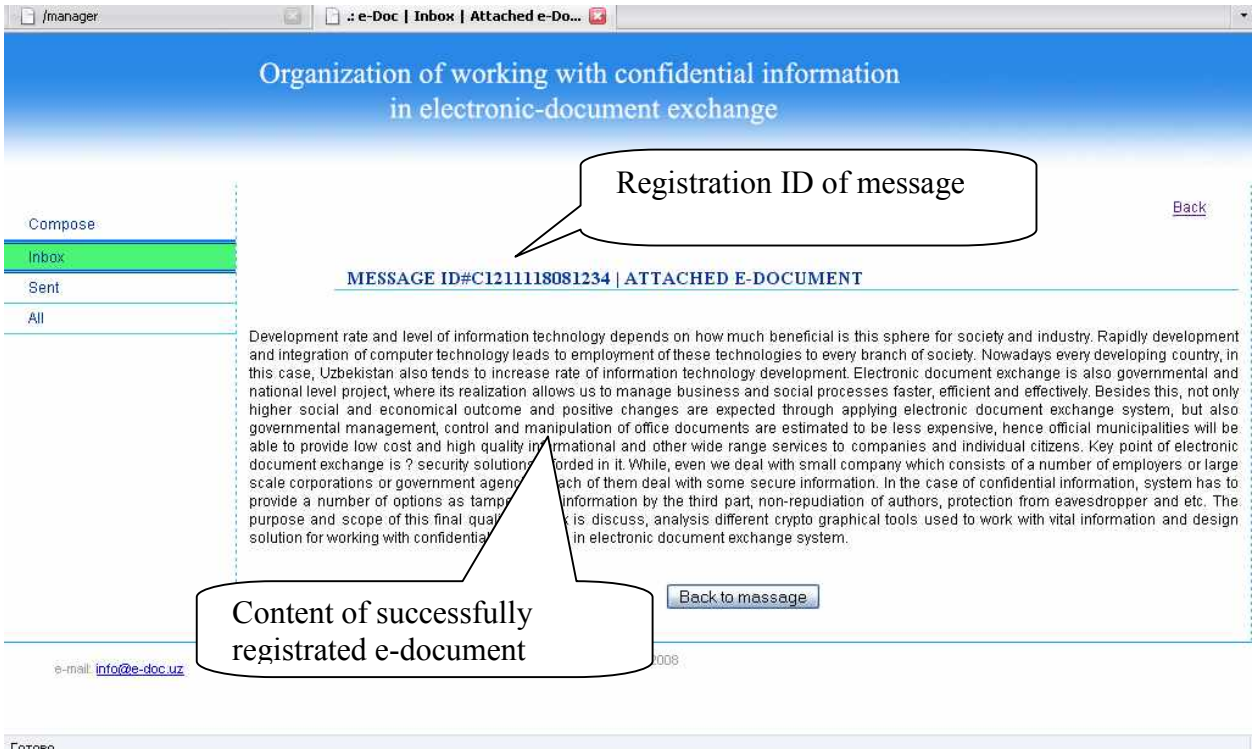


Figure 3.18. Valid digital signature

During e-document validation, it will be checked, whether if document verification is approved, it will be shown to user, on the other case document will

not be presented to user giving appropriate alert, that document content was modified or e-sign was tampered during deliver Figure 3.19.

Figure 3.19. Case of corrupted e-document

These is detailed steps of system's work process, while all steps are exactly same as described here. But it can be customized according to owner company demand.

3.1. Comparison of complexity and robustness of different crypto algorithms

Cryptographic hash function is a transformation that takes an input and returns a fixed-size string, which is called the hash value see Chapter 2.3. Hash functions use one way functions where real message presentation can not be found from its digest value, called *hash*. Table 3.5 describe some hash functions, comparing their speed of processing and number of cycles to process each byte. A *mebibyte* (a contraction of mega binary byte) is a unit of information or computer storage, abbreviated **MiB**. This is correct for MiB abbreviation.

1 MiB = 220 bytes = 1,048,576 bytes = 1,024 *kibibytes*

Table 3.5. Hash function comparison

Algorithm	MiB/Second	Cycles Per Byte
CRC-32	256	6.8
Adler-32	936	1.9
MD5	258	6.8
SHA-1	155	11.3
SHA-256	81	21.5
SHA-512	99	17.6
Tiger	217	8.0
Whirlpool	58	30.0
RIPEMD-160	108	16.1
RIPEMD-320	111	15.8
RIPEMD-128	155	11.3
RIPEMD-256	159	11.0

As we can see in Table 3.5 most popular hash functions MD5 and SHA class functions fast enough and cycles per byte are not more as RIPEMD class functions, while in our e-document exchange system we used MD5 and SHA-256.

The Data Encryption Standard (DES) is a method for encrypting information, which selected as an official Federal Information Processing Standard (FIPS) for the United States in 1976 was replaced with Advanced Encryption Standard (AES) 2002 year, since that it is wide used encryption algorithm. Table 3.6 describes comparison of symmetric crypto algorithms by speed, number of cycles per byte, time for key setup and number of used up cycles. For our project we used DES and DES-EDE3 also known as TDES, because these algorithms are in widely used. For further development different crypto algorithms can be used.

Table 3.6. Secret key (symmetric) encryption algorithms

Algorithm	MiB/Second	Cycles Per Byte	Microseconds to Setup Key and IV	Cycles to Setup Key and IV
AES/ECB (128-bit key)	99	17.7	0.248	454
AES/ECB (192-bit key)	86	20.2	0.242	443
AES/ECB (256-bit key)	77	22.6	0.312	572
Twofish	62	28.1	7.077	12950
RC6	103	16.9	2.379	4353
DES	34	51.1	6.975	12765
DES-EDE3	13	131.2	25.240	46190
IDEA	36	48.6	0.220	403
RC5 (r=16)	82	21.3	1.989	3640
Blowfish	61	28.6	61.035	111694
SKIPJACK	10	173.8	2.991	5474

A public key cryptosystem is an asymmetric cryptosystem where the key is constructed of a public key and a private key. The public key, known to all, can be used to encrypt messages. Only a person that has the corresponding private key can decrypt the message. One of the most widely used public key cryptosystem is RSA. In RSA a public key is constructed by multiplication of two very large primes. To completely break RSA one needs to find the prime factors. In practice, RSA has proved to be quite slow, especially the key generation algorithm. RSA is not well suited for limited environments like PDA's, mobile phones and smart cards without RSA co-processors, because it is hard to implement large integer modular arithmetic on such environments. RSA also requires longer keys in order to be secure. Table 3.7 compares three encryption public key also known as asymmetric encryption algorithms number of operations per millisecond.

Table 3.7. Public key (asymmetric) encryption algorithms

Operation	Milliseconds/Operation	Megacycles/Operation
RSA 1024 Encryption	0.07	0.13
RSA 1024 Decryption	1.52	2.78
LUC 1024 Encryption	0.07	0.13
LUC 1024 Decryption	2.32	4.25
LUCELG 1024 Encryption	1.86	3.40
LUCELG 1024 Decryption	1.67	3.05

Both secret key and public key cryptosystems can be used to digitally sign documents, however for public key cryptosystems are more suitable for large scale projects, while it provides better key management features. Table 3.8 compares some digital sign algorithms by operations done per millisecond and per megacycles.

Table 3.8. Public key (asymmetric) encryption algorithms

Operation	Milliseconds/Operation	Megacycles/Operation
RSA 1024 Signature	1.42	2.60
RSA 1024 Verification	0.07	0.13
LUC 1024 Signature	2.37	4.34
LUC 1024 Verification	0.07	0.13
DSA 1024 Signature	0.47	0.85
DSA 1024 Verification	0.52	0.95
ESIGN 1023 Signature	0.21	0.38
ESIGN 1023 Verification	0.07	0.12

Conclusion

For wide usage of e-document exchange system, we developed it as a web based application and for effective functionality suitable technologies were selected. As we propose our solution to different scale companies, in this project we used Open source technologies, for instance MySQL database management system as database system and Apache web server as web environment. To provide security of system and e-document exchange, special cryptographic libraries was developed, while new proposed cryptographic algorithms work with big numbers, special classes as *KattaSon*, *KattaButunSon* and *KattaKasrSon* was developed for algorithm realization.

Web based solution has to operate as we defined, so system has to provide all functionality starting from user registration to confidential e-document exchange. Different level of security can be achieved using different cryptographic algorithms, while for the same purpose we can use a number of encryption/decryption algorithms interchangeably. Here different algorithms are compared for complexity of operation and cryptographic robustness.

IV. Recommendations for employing objects and usage sphere of solution

For widely usage of electronic document management system, it has to be developed pointing to web environment. Where in application based solutions, requirements for platform and operation system are very specific and usually it is difficult to have such environment for small company or nation level interests, due to technical or financial obstacles. While solution, developed in this final qualifying work is based on the web, hence it enables vital information exchange through public network. Chapter 4.1 and 4.2 are devoted to implementation of constructed solution in different objects, such as small and large companies, where secure document exchange through internal and external network is compulsory.

Applying to small companies has wider opportunity, because independent person can choose rich library of algorithms for private usage, while government agencies and other national level organizations are not able to use all encryption and digital sign algorithms, because of crypto algorithm's export restriction outside of country. Of course there are some public crypto algorithms which can be used without any licensing, while these algorithms are not patented in any country. On the other hand to easily implement without licensing and checking for robustness of cryptosystem, country has to provide national crypto algorithms, which can be used by every company inside country.

This and other aspects of project implementation are discussed in Chapter 4.1 and Chapter 4.2.

4.1. Solution for companies and large corporations

There are a lot of benefits to employ e-document exchange system to small and large business companies, while bigger corporations will be able to faster exchange e-documents with each other. On the other hand, as scope of company is small, there is less demand for secure exchange e-document and if there is small number of system's users there will not be real requirement for secure exchange of e-documents. However, as scope of company is large, real requirement for e-documents growth and there more demand for secure document exchange system.

Let's consider some example. First of all, e-document exchange can be tested in small business applications, while as an example we will see case of some university which has about 5000 employers. There are 10 faculties, where each faculty has on average five different departments and as they are sub departments of the same university, they have to exchange e-documents with each other and there is additional e-document traffic flow between responsible departments and central management system. Of course in such organization confidential authorities use some e-documents which have to be exchanged only between source and destination points (e.g. these can be exchange of e-document between head of departments and university managers), as document exchange is going between different departments, there has to be central registration office, where each e-documents has to be registered with unique ID associated with meta information about actual e-document and date of recipient and delivery. On the other hand, solution which is recommended here can be used to easily supply in different level of confidentiality using special cryptographic algorithms and applying different key bit length of public/private and secret key encryption. Our provided example offers simple format of e-documents which can be easily configured depending on company's requirement. The main thing which has to be exactly provided is internal network support, suitable operation system for user terminals and central server which hosts web application and database.

4.2. Using confidential information exchange solution in governmental projects

To widely implement e-document exchange system, it has to be supported in most environments, while in this case most optimal decision is web based application (see Chapter 2.2). Because it has more flexible and extensible features and it is easy for further development.

To employ e-document exchange system in government, solution it has to have most effective functionality and suitable user interface and as government is multi level management system, document has to be multi level too (see Chapter 1.4). Here, depending on confidentiality level, applied security tools may vary and if e-document exchange happens in national level, here cryptographic tools also has to be national, while each government's cryptographic tools are protected by international rights and some of them is restricted to export, while for widely usage country has to produce its own national standards.

One more option for secure e-document exchange system is – key management. As we mentioned in the Chapter 3.2 to sign document with secret key cryptography, each user had to agree key pair with each other and of course in multi level government management system accomplishment of this require is unfeasible. Of course the best solution here is building PKI (Public key Infrastructure), where all users will have their public/private key pair and people in different level of society will be able to e-documents using their public key pairs. Additional issue, there is not only one solution as document exchange system is exists and used, but there are a number of different solutions are used in a market. So there has to be central PKI system which serves as central public key registration and revocation centre. Nowadays every country going to build such central servers (see chapter 1.3) and in Uzbekistan current day there are three public key certification centers and each of them can collaborate, as was shown in presentation /12/ one countries certification center's issued public key can be used to sign e-document and other country's certification centre is able to verify that

digital sign and do appropriate procedures of e-document management, registering it in central storage and giving cheque about e-document acceptance.

In our solution we used secret key encryption to digitally sign e-documents and verify it. RDBMS was used to keep secret key pair, while this demo solution can be extent to verify digital sign of e-documents with public/private key pair from certification center. All important steps are connecting to certification centre and giving public key of user, with its certification. Applying this step we will be able to implement our solution to government agencies. As now digital sign e-document with secret keys is provided.

Conclusion

Developed solution can be used in different scale project and as we developed and proposed main operational functionality in the scope of our project, this solution can be implemented in different scale. For instance, small companies can use our solution separating web server for central registration office and constructing local area network for user connection, thus employers will be able to confidentially exchange documents.

On the other hand as company is small and there no much stuff, real demand for the secure document exchange will not be so great, whilst as scale and number of stuff increases, security of vital information exchange also increases proportionally.

The main target of e-document exchange system is government agencies; while here is multi level management and lot of inner and outer attacks to existing e-document exchange system. Our proposed model can be extent to large scale projects also, while main necessary functions, such as encryption and digital signature are already employed and tested on developed exchange system.

V. The safety precautions

Chapters devoted to discuss technical and personal safety at the time of working with computer and personal health care. While all technical processes require dealing with computer, user and administrator have to follow these rules. In Chapter 5.1 we'll talk about safety requirements working on personal computer, where discussions about switching on and off technical devices are illustrated. In the Chapter 5.2 guide for Ergonomics – which means personal indication to work on computer is given. Finally we'll finish chapters giving appropriate conclusions.

5.1. Safety requirements working on personal computer

Student have to ask permission before switching on personal computer from laboratory assistant. Accordingly laboratory assistant has to make sure that all tools, such as cables, natural and artificial in/out ventilation, workspace brightness and etc are working correctly.

Before **turning on computer student must**:

- Prepare workplace, and remove all unnecessary things;
- Inform laboratory assistant about defect, and it is denied to use broken sockets;

During work process student has to follow:

- Don't use broken personal computers or its tools;
- Don't turn of and turn on tools without permission;
- Don't work in badly lighted or ventilation workplace;
- Don't open case of any device;
- Don't turn on printers, scanners and any other equipment if not necessary;
- Don't leave computer or other device without checkup;

Requirements after finishing work for students:

- Close all application;
- Switch of user and turn off computer;

- Remove cables form electricity;

Requirements after finishing work for laboratory assistant:

- Check laboratory rooms for any sign of fire;
- Check room for strange things;
- Switch off all lights, air conditioners and other electrical devices;
- Close windows, doors, stamp the room; turn on signaling if exists;
- Give key and room to security service and put sign about this to registration journal;

5.2. Ergonomics – personal indication in safety in workplace

Ergonomics is a science concerned with the 'fit' between people and their work. It takes account of the worker's capabilities and limitations in seeking to ensure that tasks, equipment, information and the environment suit each worker.

To assess the fit between a person and their work, ergonomics consider:

- the job being done and the demands on the worker;
- the equipment used (its size, shape, and how appropriate it is for the task);
- the information used (how it is presented, accessed, and changed).

There are five aspects of ergonomics, safety, comfort, ease of use, productivity/performance, and aesthetics. Based on these aspects of ergonomics, examples are given of how products or systems could benefit from redesign based on ergonomic principles.

1. Safety - Medicine bottles: The print on them could be larger so that a sick person who may have impaired vision (due to sinuses, etc.) can more easily see the dosages and label. Ergonomics could design the print style, color and size for optimal viewing.
2. Comfort - Alarm clock display: Some displays are harshly bright, drawing one's eye to the light when surroundings are dark. Ergonomic principles could redesign this based on contrast principles.

3. Ease of use - Street Signs: In a strange area, many times it is difficult to spot street signs. This could be addressed with the principles of visual detection in ergonomics.
4. Productivity/performance - HD TV: The sound on HD TV is much lower than regular TV. So when you switch from HD to regular, the volume increases dramatically. Ergonomics recognizes that this difference in decibel level creates a difference in loudness and hurts human ears and this could be solved by evening out the decibel levels. Voicemail instructions: It takes too long to have to listen to all of the obvious instructions. Ergonomics could address this by providing more options to the user, enabling them to easily and quickly skip the instructions.
5. Aesthetics - Signs in the workplace: Signage should be made consistent throughout the workplace to not only be aesthetically pleasing, but also so that information is easily accessible for all signs.

Conclusion

Technical and personal safety is one of the important task in our life, where nowadays every person is dealing with high voltage devices such as personal computers. In this chapter we explored technical safety both for student and laboratory assistant, looking overall following rules to deal with computers. Also talked about ergonomics – which means personal indication in safety in workplace, looking to overall criteria to work in safe and productive workplace.

Final conclusion

To develop technologies as a suitable for all business and government departments there was proposed standard structure, format and content for e-documents. Now widely available three such standard formats, first “MoREQ” was developed by IDA (Interchange of Data between Administrations) Programme of the European Commission, second “DoD 5015.02-STD, Electronic Records Management Software Applications Design Criteria Standard” by Department of Defense United States and last “JSR 170, Standardizing the Content Repository Interface” an open Source technology targeting worldwide usage. All of these three standards are revised in the content of appropriate chapter.

As paperless office services increase and society moves to paperless, governments also will be able to provide its services over network, while this leads to build e-government of each country. Now e-Government project of each country is fulfilling continuously, taking account all technological improvements.

Purpose and scope of this final qualifying work is discuss, analysis different cryptographic tools used to work with vital information and design solution for working with confidential information in electronic document exchange system.

To provide rich services available in e-document exchange systems it has to be extensible and optimal for manipulation and storage. Structure of e-document has to be extensible in multi level government management system, where in this purpose nowadays’ wide technology is XML format. As XML format is portable to transfer through network, it can be used to construct complex structure of e-documents. Rich structure of XML document is constructing by W3C organization, while organization is intensively working to provide encryption, digital sign inside same XML document. In this purpose, XML Signature and XML Encryption was developed and proposed.

Fully implementation if e-document exchange can be gained if core application is able to support those options, while in our case core technology is database management system. To provide confidential information flow in e-

document exchange system, we developed 10 database tables, while each of them has appropriate operational functionality. Beside database architecture exact steps of e-document flow is defined, taking in account all exceptional cases.

Confidential information exchange can be achieved applying cryptographic tools, whilst in this project we used four major cryptographic tools – encryption, hash function, digital signature and data compression and we applied new encryption algorithm, which is hybrid cryptosystem – generating encryption key via asymmetric algorithm and symmetric encryption algorithm for further data exchange.

For wide usage of e-document exchange system, we developed our solution as web based application and for effective functionality suitable technologies are selected. As we propose our solution to different scale companies, in this project we used Open source technologies, for instance MySQL database management system as DBMS and Apache web server as web environment. To provide security of system and e-document exchange, special cryptographic libraries was developed.

Different level of security can be achieved using different cryptographic algorithms, while for the same purpose we can use a number of encryption/decryption algorithms interchangeably. Here different algorithms are compared for complexity of operation and cryptographic robustness.

Developed solution can be used in different scale project and as we developed and proposed main operational functionality in the scope of our project, this solution can be implemented in different scale. For instance small companies can use our solution separating web server for central registration office and constructing local area network for user connection, thus employers will be able to confidentially exchange documents. Our proposed model can be extent to large scale projects also, while main necessary functions, such as encryption and digital signature are already employed and tested on developed exchange system.

References:

1. Laws of Uzbekistan: UzR 29.04.2004 y. 613-II.
   - On electronic commerce;
   - On electronic document flow;
2. Laws of Uzbekistan “On digital signature”: 11 December 2003 year
3. National standard of Uzbekistan O’z DSt :2004 – Organization cryptographic protection of information in Uzbekistan Republic
4. Нир Вулкан. Электронная коммерция. М.: «Интернет-трейдинг», 2003, - 296 с
5. Юрасов А.В. Электронная коммерция. Учебное пособие.М.:Дело, 2003, - 480 с.
6. Bruce Schneier, *Applied Cryptography*, John Wiley & Sons, New York, 1994
7. A. Salomaa, *Public-Key Cryptography*, Springer-Verlag, 1990
8. S. Barichev, *Криптография без секретов*.
9. The Information Age, Emmanuel C. Lallana, Ph.D with assistance from Margaret N. Uy, may 2003, e-nSEfM Task Force IMDP-nPDIP
10. Khamdamov R.Kh., Ergashev A.K. *Решение задачи о рюкзаке методом обобщенных неравенств*, Сб. научных трудов ТашГТУ, 1993.
11. International Conference on Information Technology Promotion in Asia 2007 (ITPA-2007), *Private box algorithm,* Rustam Kh. Khamdamov, Nodir Kh. Kodirov, September 24-25, 2007, Tashkent, Uzbekistan
12. “Development problems of electronic document management infrastructure and services and topical problems of Digital Signature Use”, April 23-25, 2008, Tashkent, Republic of Uzbekistan
13. Electronic Document (Data) Exchange Services, Chad Corneil Founding Member, ASP Industry Consortium, BUSINESS BRIEFING: GLOBAL PURCHASING AND SUPPLY CHAIN STRATEGIES

14. Elisa Bertino, Barbara Carminati, and Elena Ferrari, “XML Security,” Information Security Technical Report 6, no. 2 (2001): 44-58.

15. Deputy Assistant Secretary of Defense for Networks and Information Integration Specification, “Department of Defense Discovery Metadata Specification (DDMS), Version 1.3,” July 29, 2005

Internet sources:

16. Interagency Electronic Document Exchange, Research, Development and Evaluation Commission, Executive Yuan, http://www.rdec.gov.tw/fp.asp?xItem=13673&ctNode=8666

17. A Web-Based Model for Electronic Document Exchange, By Durno, John, Publication: Library Trends, January 1, 2006, http://www.allbusiness.com/information/information-services-libraries-archives/1009662-1.html

18. XML Encryption (W3C XML Encryption Working Group, 2001) – www.w3.org/Encryption/.

19. DeRose, S. and J. Clark, eds. XML Path Language (XPath), Version 1.0. W3C Recommendation, 16 November 1999. http://www.w3.org/TR/xpath

20. W3C Technical Architecture Group. Architecture of the World Wide Web, Volume One. W3C Recommendation, 15 December 2004. http://www.w3.org/TR/webarch/

21.http://www.jcp.org/ – The Java Community Process(SM) Program

22. Fielding, Roy Thomas. Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine, 2000. http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

23. Day Software. Content Repository API for JavaTM Technology Specification. Java Specification Request 170, version 0.16.1, 24 December 2004. http://www.jcp.org/en/jsr/detail?id=170

24. Sun Microsystems. Java Transaction API (JTA) Specification. http://java.sun.com/products/jta/index.html

25. IBM. Workspace Versioning and Configuration Management (WVCM) API. Java Specification Request 147. http://www.jcp.org/en/jsr/detail?id=147

26. NIST.gov - Computer Security Division - Computer Security Resource Center http://csrc.nist.gov/groups/ST/hash/sha-3/index.html

27. http://www.netcraft.com/ – Netcraft Web Server Survey

28. www.wikipedia.org – Wikipedia, the free encyclopedia

29. www.nist.gov – National Institute of Standards and Technology

30. www.itl.nist.gov/fipspubs/ – Federal Information Processing Standards Publications

31. http://www.moreq2.eu/ – MoReq2 - the next generation of the Model Requirements for Electronic Records Management

32. www.obi.giti.waseda.ac.jp/e_gov/ – The Waseda University Institute of e-Government

33. http://www.mysql.com/ – MySQL; The world's most popular open source database

34. http://www.apache.org/ – The Apache Software Foundation

Appendix 1

Abbreviations and Acronyms

ASCII	American Standard Code for Information Interchange
BIFF	Binary Interchange File Format
CFR	Code of Federal Regulations
CSS	Cascading Style Sheets
DBMS	Database Management System
DTD	Document Type Definition
E-mail	Electronic mail
EO	Executive Order
EXIF	Exchangeable Image File Format
FIPS	Federal Information Processing Standard
GIF	Graphic Image Format
HTML	Hyper Text Markup Language
ISO	International Organization for Standardization
IT	Information Technology
JPEG	Joint Photographic Experts Group
LAN	Local Area Network
PDF	Portable Document Format
PKI	Public Key Infrastructure
PNG	Portable Network Graphics
RMA	Records Management Application
SGML	Standard Generalized Markup Language
SMTP	Simple Mail Transfer Protocol
SVG	Scalable Vector Graphics
TCP/IP	Transmission Control Protocol/Internet Protocol
TIFF	Tagged Image Interchange Format
URL	Uniform Resource Locator

UUID	Universally Unique Identifier
W3C	World Wide Web Consortium
WAN	Wide Area Network
XHTML	eXtensible Hypertext Markup Language
XML	eXtensible Markup Language
XSLT	eXtensible Stylesheet Language Transformations
XUL	XML User-interface Language

Tashkent University of Information Technologies

e-Commerce department

Organization of working with confidential information in electronic-document exchange

Applicant: Kodirov N.Kh.
Supervisor: prof. Khamdamov R.Kh.

Content

- Overall status and analysis of electronic document exchange
- Constructing structure of electronic document exchange system
- Confidential information in electronic-document exchange solution
- Employing objects and usage sphere of solution
- Conclusion

Overall status and analysis of electronic document exchange

- e-document exchange system - advantages and obstacles
- International standards, formats and proposals
- Purpose, scope and expected results of final qualifying work

e-document exchange system, advantages and obstacles

- quick and easy;
- scalability of costs and technology;
- leverage the Internet to facilitate the flow of information between organizations
- high-quality, high-efficiency e-government services;

International standards, formats and proposals

- Model Requirements for the Management of Electronic Records

- Electronic Records Management Software Applications Design Criteria Standard. DoD 5015.02-STD;

- JSR 170;

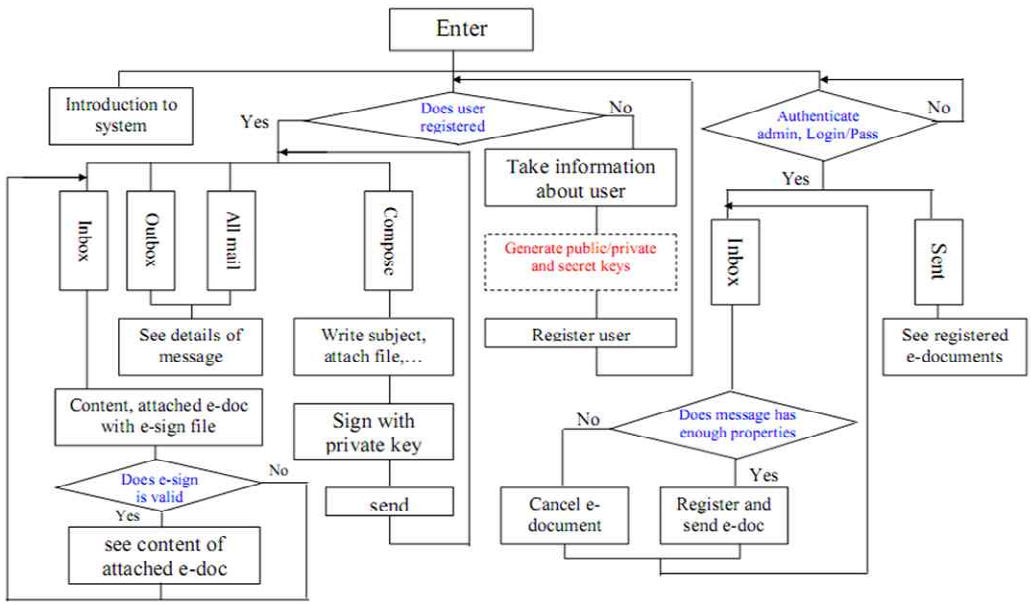
Purpose, scope and expected results of final qualifying work

The purpose and scope of this final qualifying work is discuss, analysis different cryptographic tools used to work with vital information and design solution for working with confidential information in electronic document exchange system.

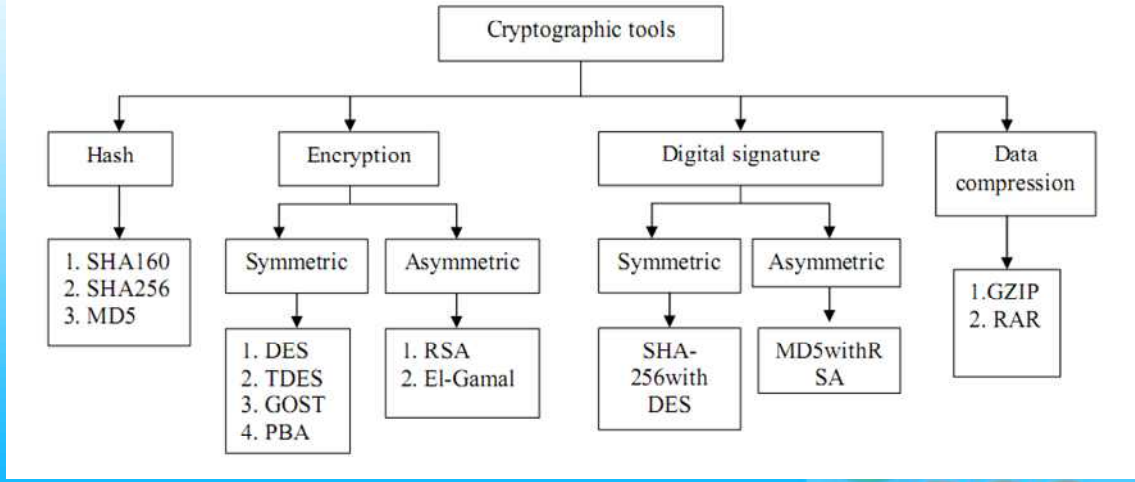
Electronic document exchange system structure

- architecture of electronic document exchange system;
- cryptographic tools on working with confidential information;
- new cryptographic algorithm in working with confidential information (PBA);

Architecture of electronic document exchange system



Cryptographic tools on working with confidential information



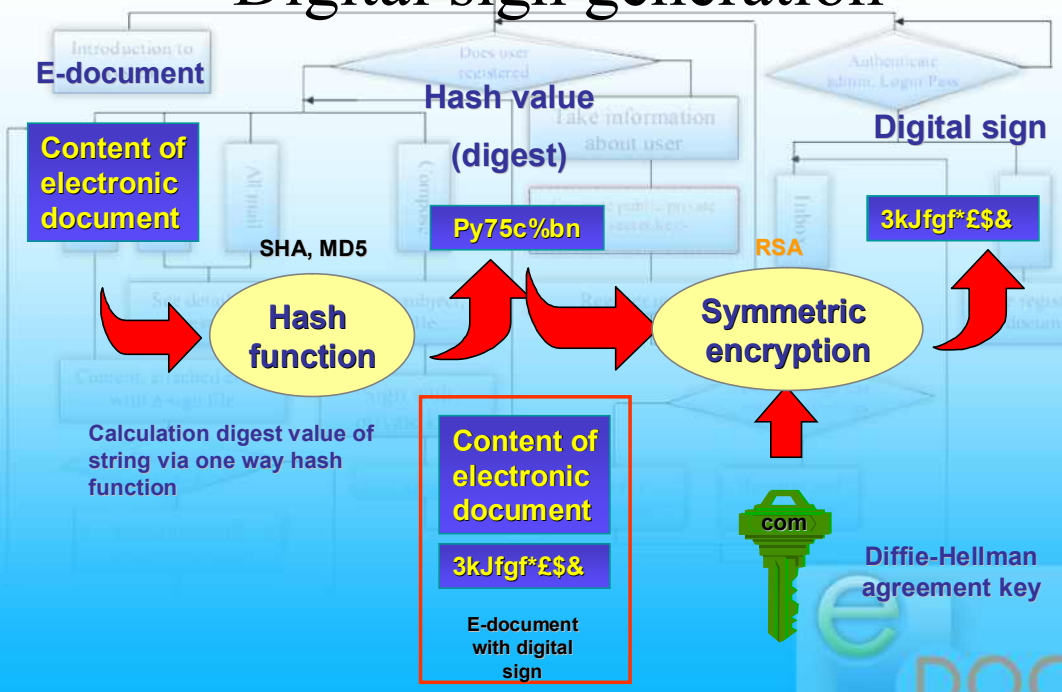
Cryptographic tools on working with confidential information

Name	Package	Summary of class
SHA-256	nodir.crypto	Evaluates hash value (digest) of string, where length of digest is 256 bits. An algorithm realizes all steps proposed in FIPS PUB 180-3 specification.
KattaSon	nodir.crypto	Does calculation of big numbers, which are from 200 to 800 digits in length.
KattaKasrSon	nodir.crypto	Manipulates decimal number encapsulated by bytes. Has method for addition, remain, modular division and etc. works same as core java.math.BigDecimal class.
KattaButunSon	nodir.crypto	Does calculation of big integers and has all basic mathematic calculations, including cryptographic methods. Actually, private analog of java.math.BigInteger class. Object encapsulated as array of bytes.
DiffieHellman	nodir.crypto	Used to generate secret key pair, defined in Diffie-Hellman key agreement cryptosystem.
DESEncryption	nodir.crypto	Does encryption and decryption of message, returning byte representation of cipher. Method realizes steps defined in FIPS PUB 46 “Data Encryption Standard” specification. As development tool I used third party library provided from Bouncycastle free library.
Arrays	nodir.java.util	Manipulates arrays, which elements are *KattaButunSon* and *KattaKasrSon*, sorting them as required in *YSAUmumiy* algorithm. Method uses fast binary sort algorithm to rank elements in ascending order.
YSAUmumiy	nodir.crypto	Class which realizes Private Box Algorithm. Used to encryption/decryption of messages.

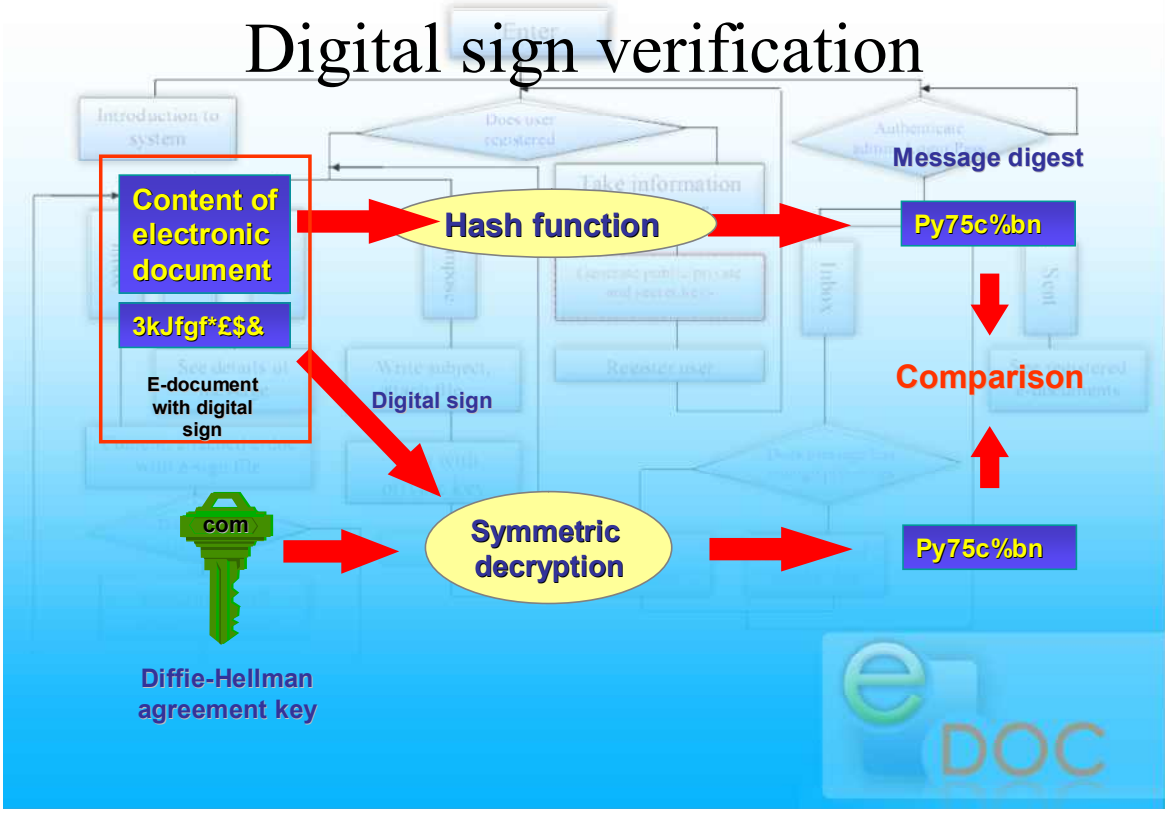
New cryptographic algorithm in working with confidential information

- **Private Box Algorithm** (PBA) based on rucksack problems, which were first introduced in 1979 by Ralph Marklin and Martin Hellmann.
- algorithm use rucksack systems and exchanged by elements through public channel after using modular arithmetic calculation for each element
- rucksack elements generate separately by each abonent on the basis of private parameter
- Pseudorandom numbers are used to generate private keys, which have predefined pattern

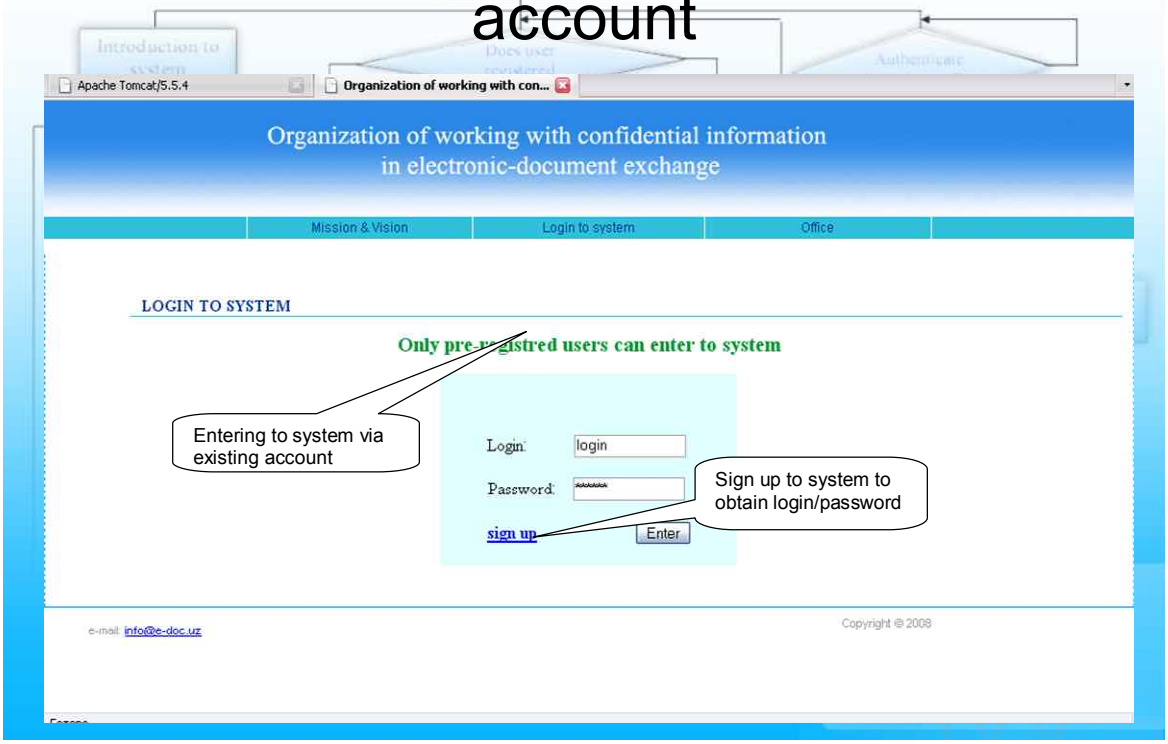
Digital sign generation



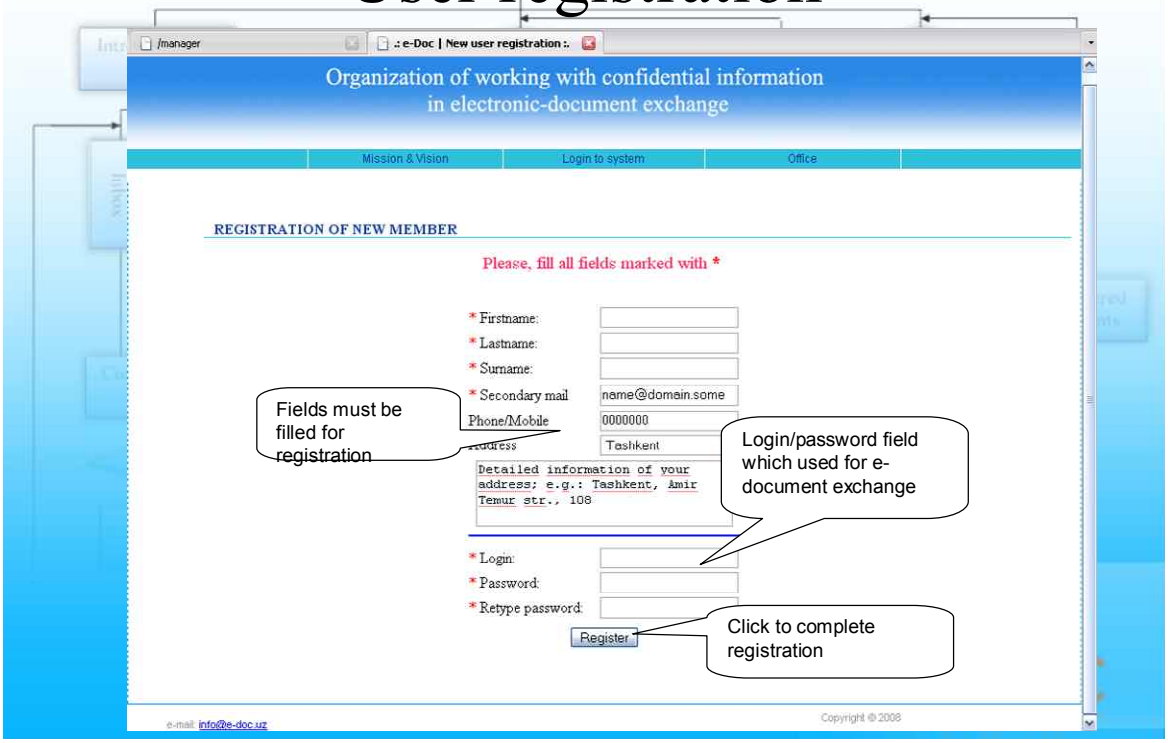
Digital sign verification



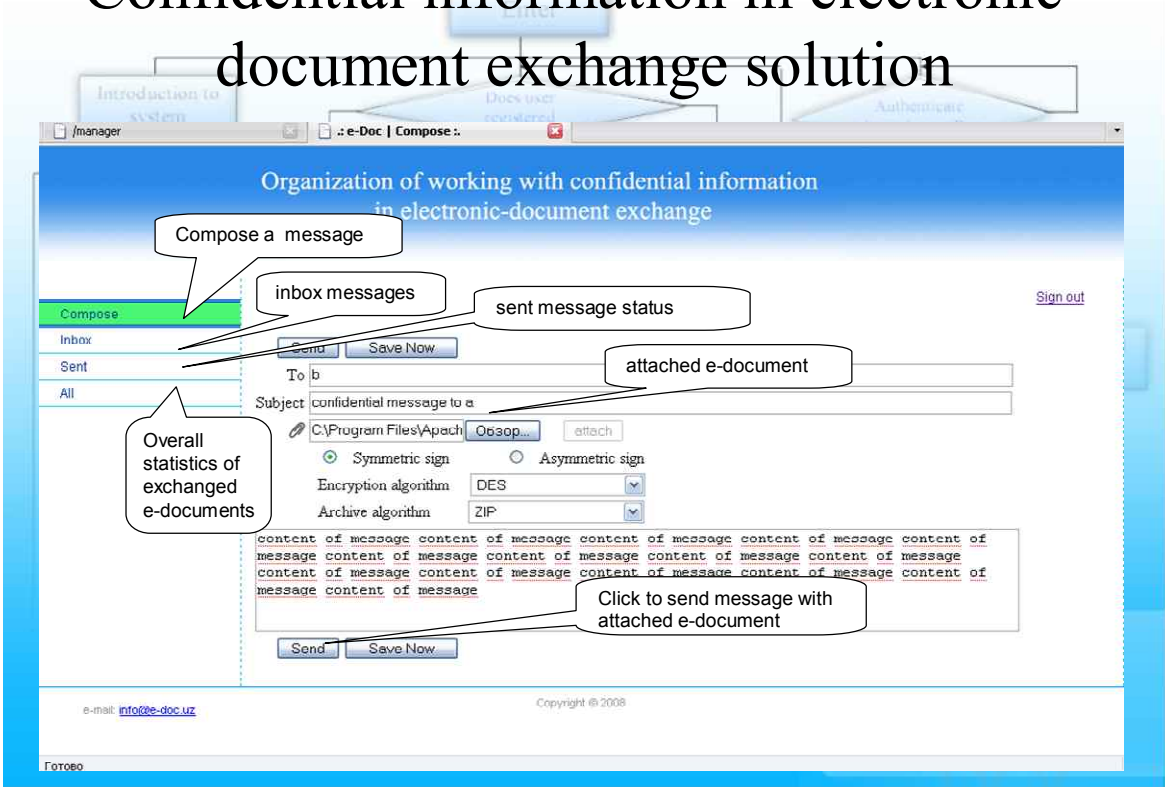
Enter to system with pre-registered account



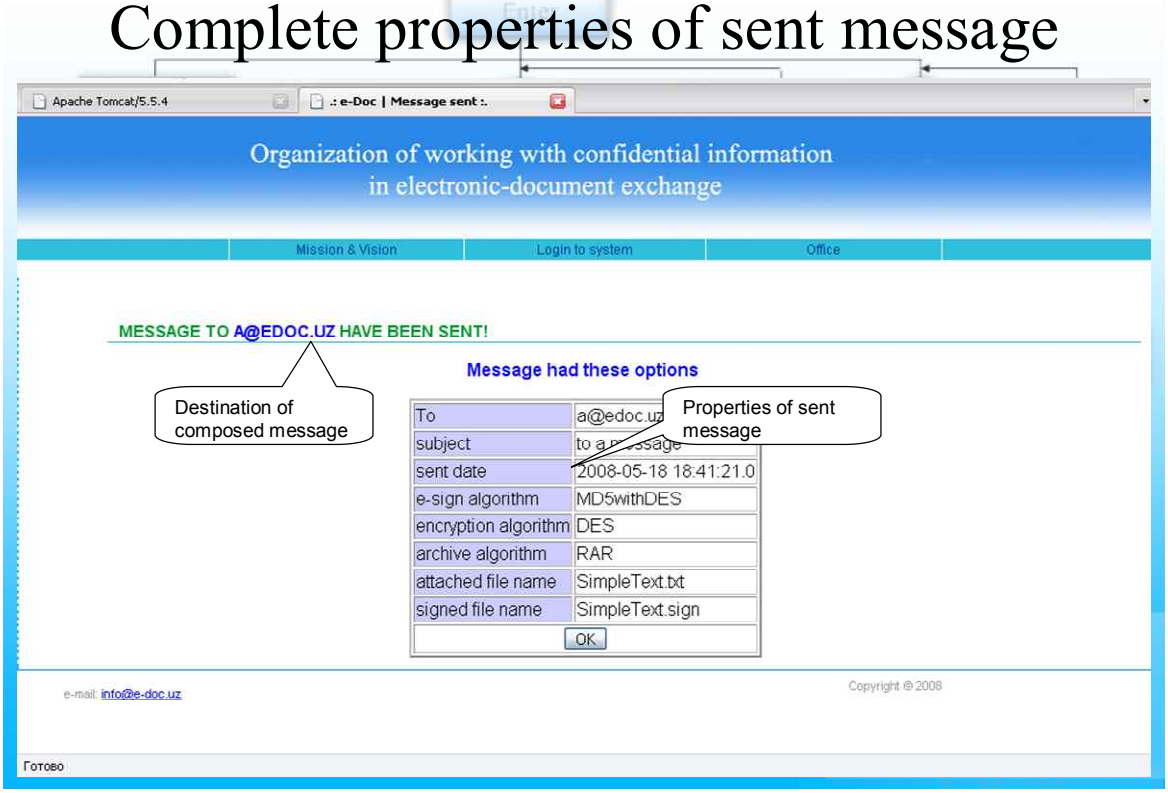
User registration



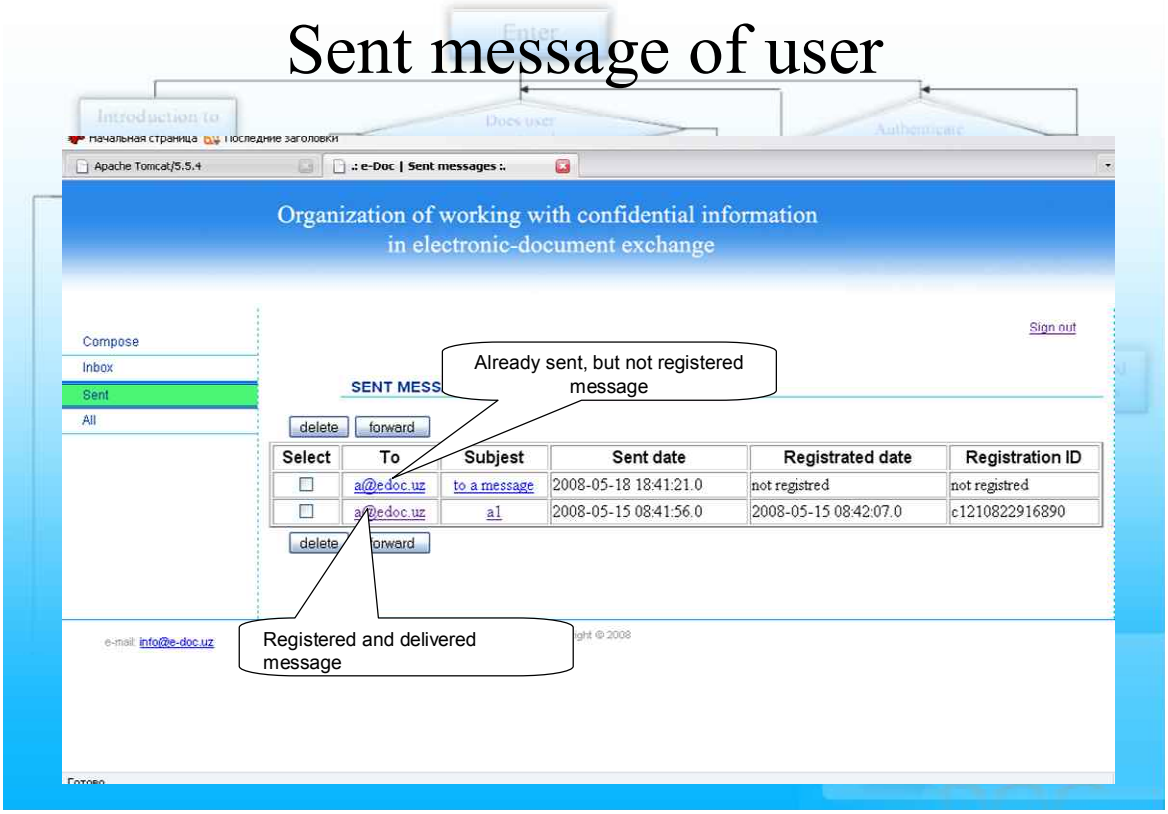
Confidential information in electronic-document exchange solution



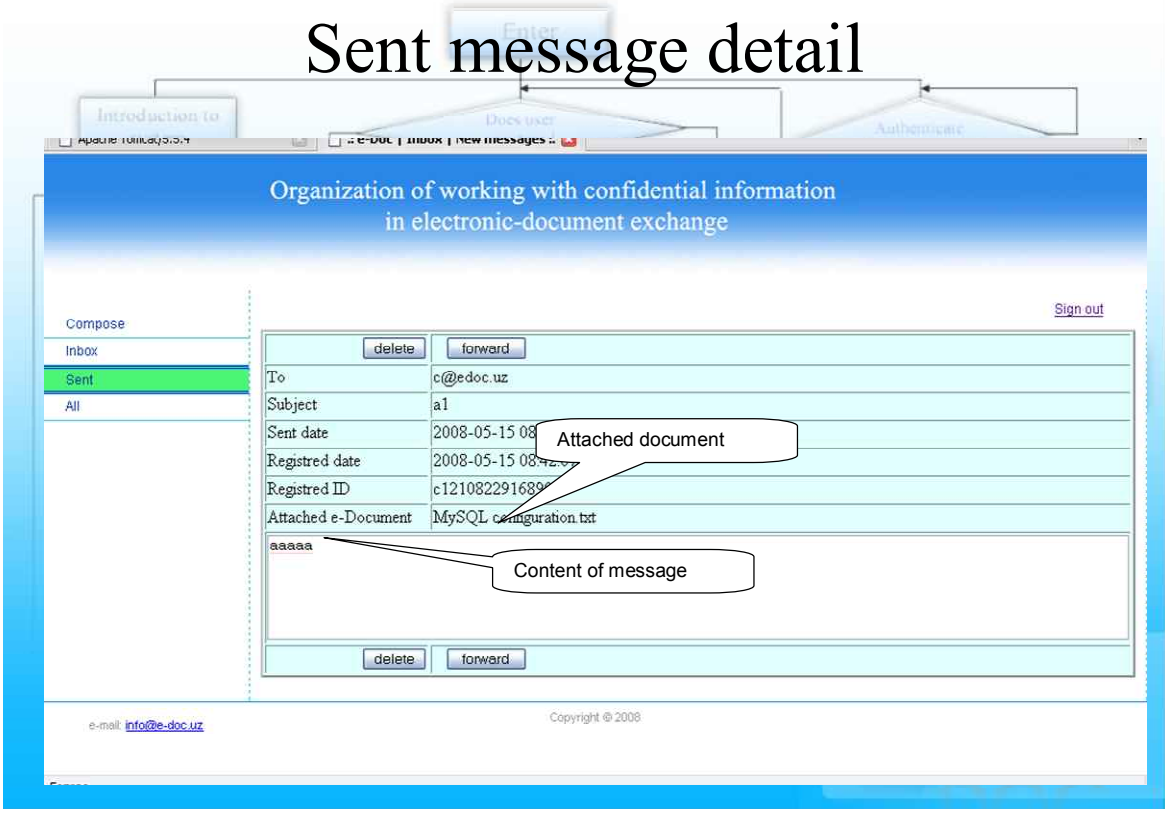
Complete properties of sent message



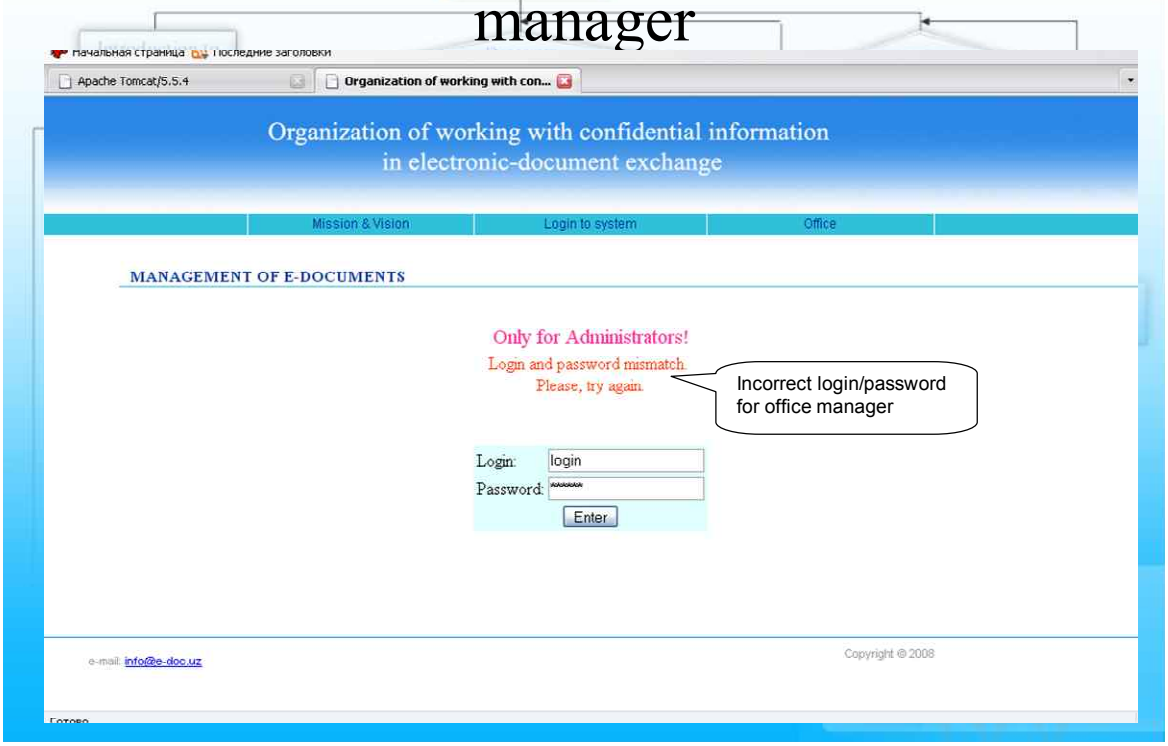
Sent message of user



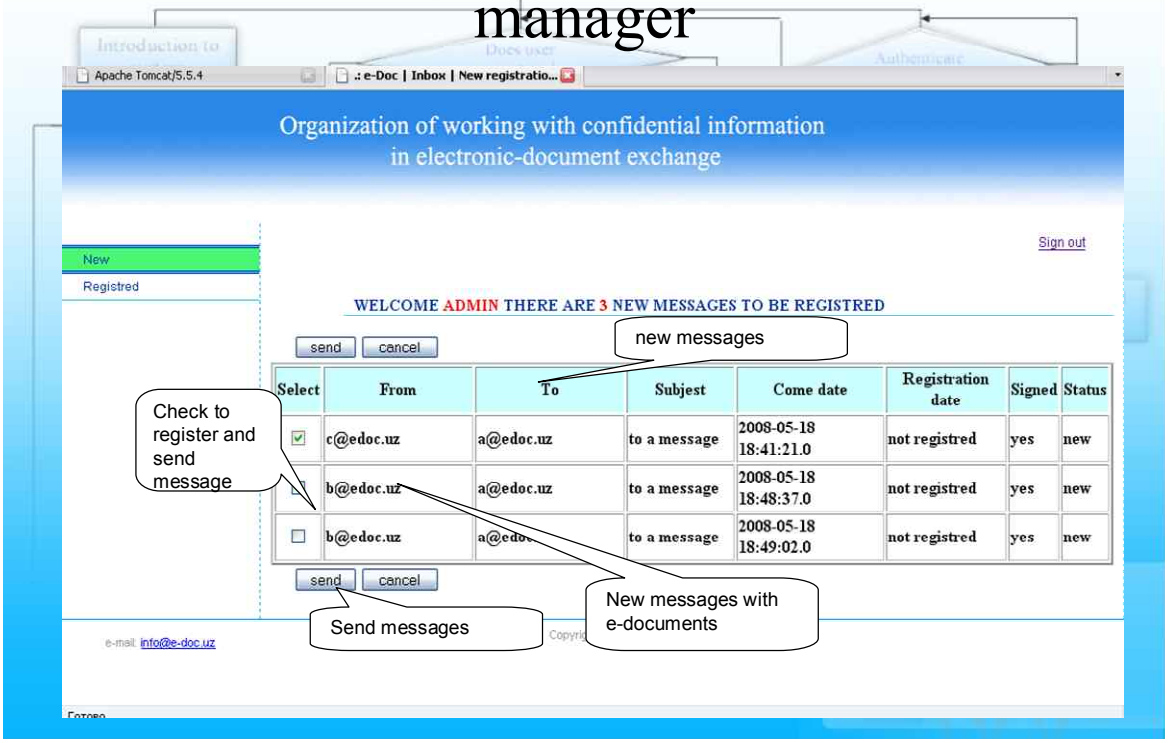
Sent message detail



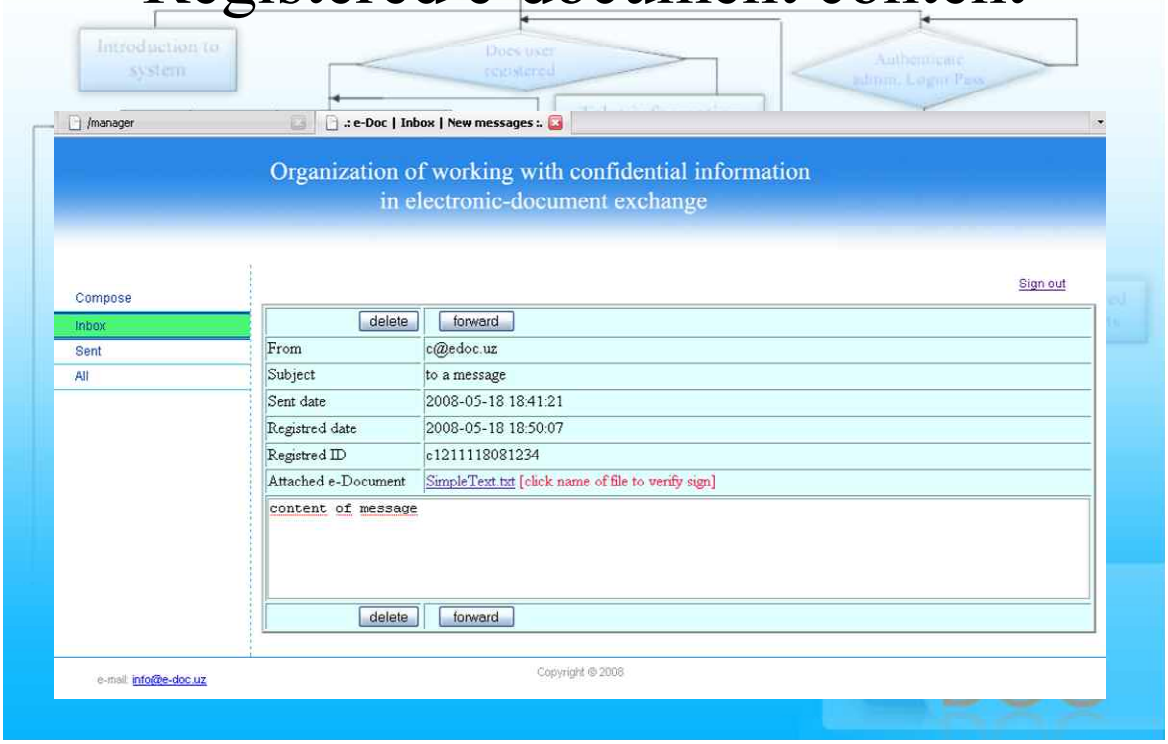
Login page of registration office manager



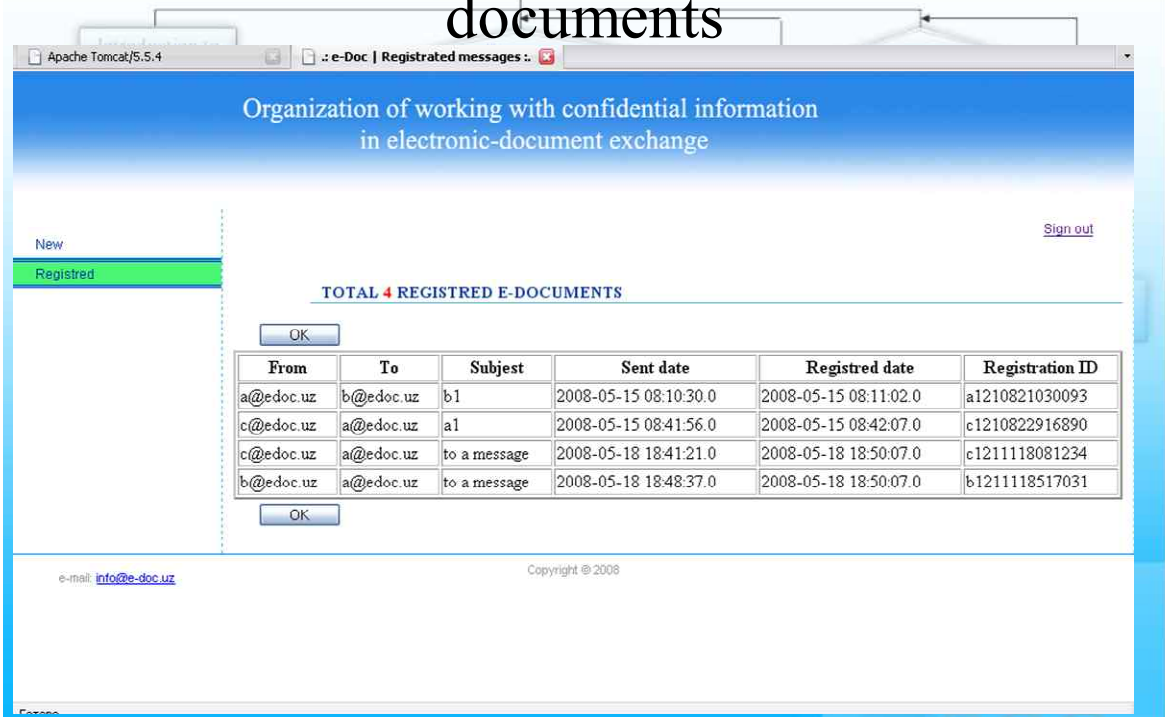
Registration of messages by office manager



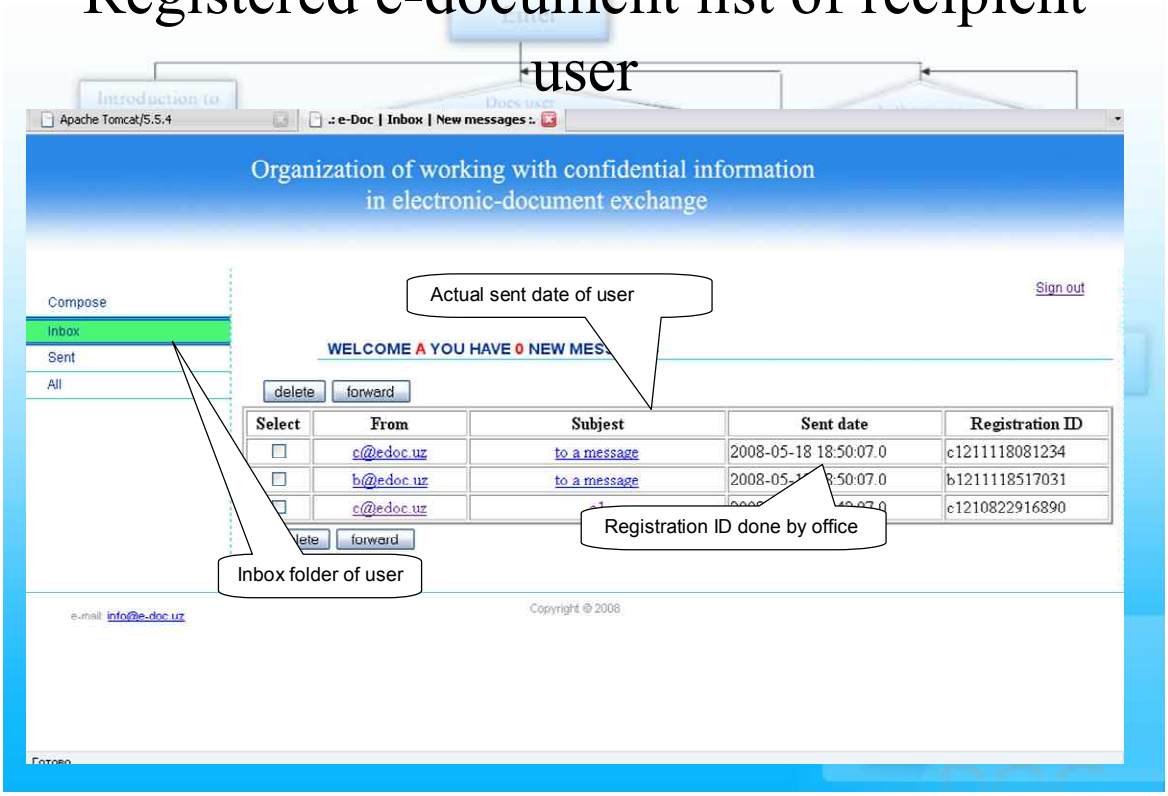
Registered e-document content



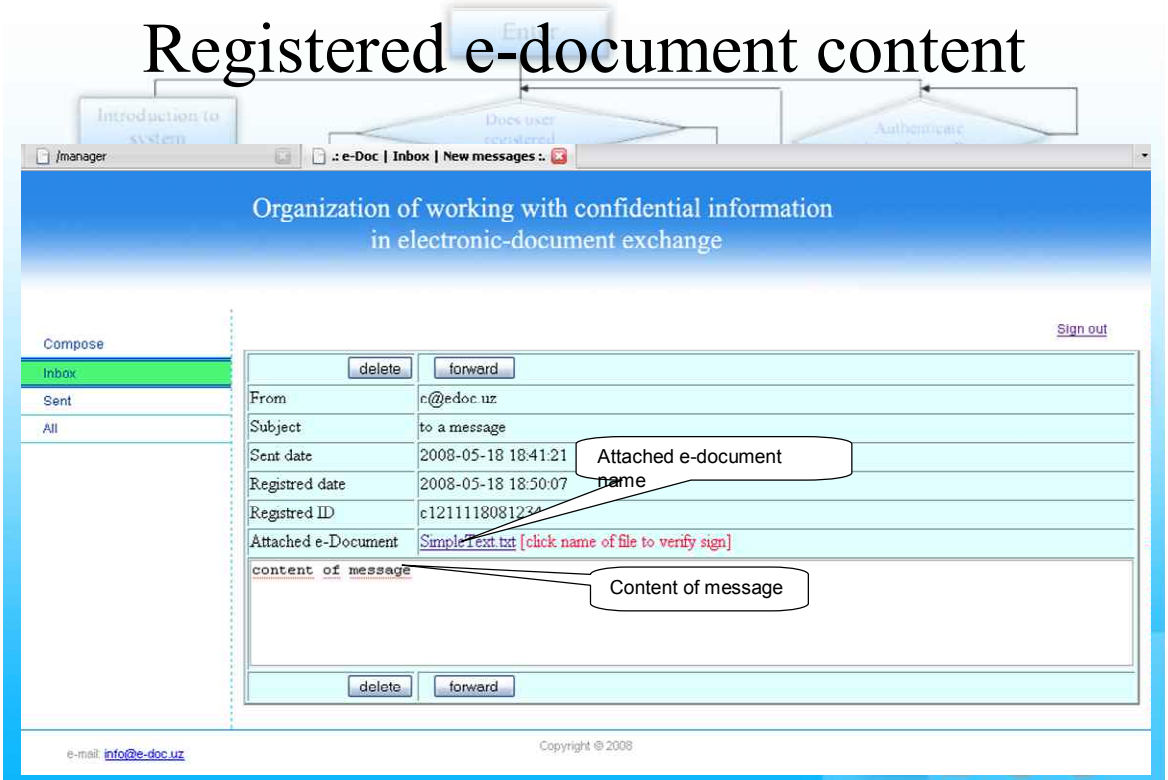
Previously registered messages with e-documents



Registered e-document list of recipient user

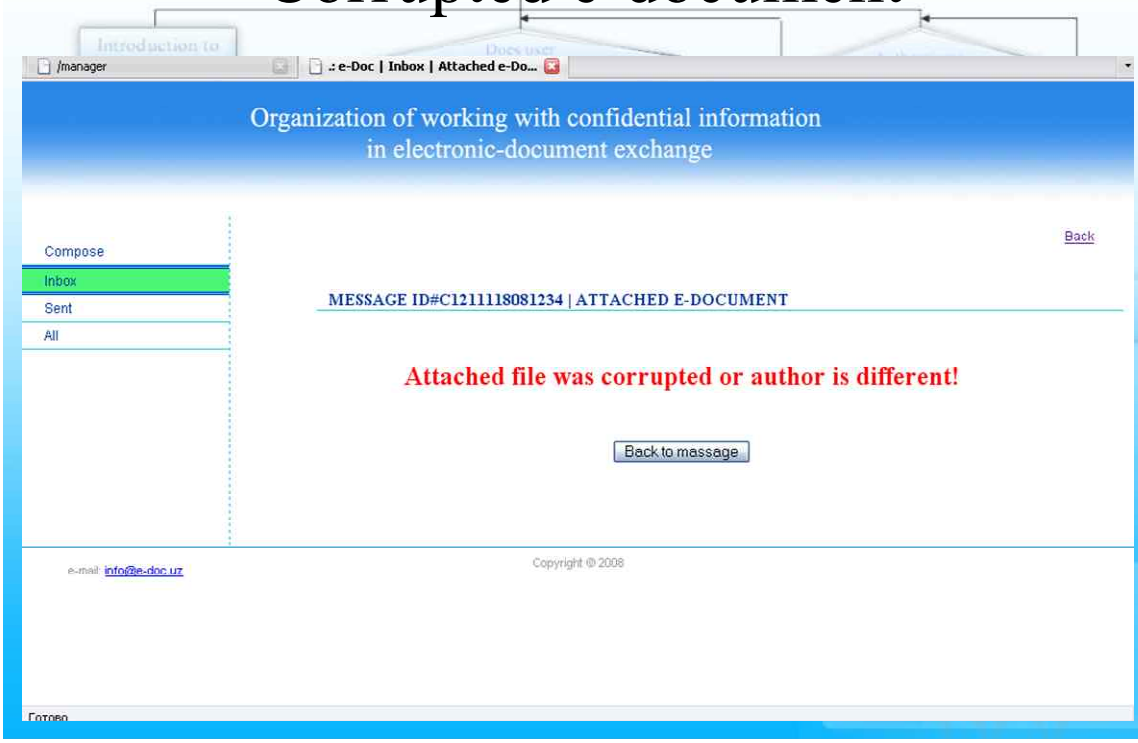


Registered e-document content



Valid digital signature

Corrupted e-document



Employing objects and usage sphere of solution

- Solution for companies and large corporations
- solution in governmental projects

Conclusion

- Standard structure, format, content for e-documents;
- XML Signature and XML Encryption can be used to digital sign inside same XML document;
- Appropriate cryptographic tools for encryption, hash function, digital signature creation and verification has to be fully implemented;
- web based application and high performance technologies has to be in use;
- Developed solution can be used in small and large corporative companies, also in governmental projects customizing to their requirements;

Thanks for attention!

Q&A…

Class KattaButunSon

```
package nodir.crypto;
import java.util.*;
public class KattaButunSon {
        private byte[] byteView;
        public int length;
        public static final KattaButunSon ZERO =
          new KattaButunSon("0");
        public static final KattaButunSon ONE =
          new KattaButunSon("1");
        public static final KattaButunSon TWO =
          new KattaButunSon("2");
        public static final KattaButunSon TEN =
          new KattaButunSon("10");
        public boolean negate;
        public KattaButunSon(){
          this("0");
        }
        public KattaButunSon(int i){
          this(String.valueOf(i));
        }
        public KattaButunSon(long l){
          this(String.valueOf(l));
        }

        public KattaButunSon(String strValue) {
          strValue = correct(strValue);
          if (strValue.equals("")){
            System.out.println("OK");
            byteView[0]=0;
          } else {
            StringBuffer stB = new StringBuffer(strValue);
            if (stB.charAt(0)=='-'){
                    negate = true;
                    stB.deleteCharAt(0);
            }
            this.length = stB.length();
            byteView = new byte[this.length];
            try{
                    for (int i=0; i<length; i++){
                            byteView[i] = Byte.parseByte(
                                    String.valueOf(stB.charAt(i)));
                    }
            } catch (NumberFormatException e){
                    System.out.println("Iltimos, to'g'ri son kiriting");
            }
          }
        }
        public KattaButunSon(byte[] arr){
          this.byteView = arr;
```

```
    }

    public byte[] getBytes(){
      return byteView;
    }
    public static KattaButunSon valueOf(String strValue){
      KattaButunSon kts = new KattaButunSon(strValue);
      return kts;
    }
    public String toString(){
      StringBuffer stb = new StringBuffer();
    for (int i=0; i<length; i++)
      stb = stb.append(String.valueOf(byteView[i]));
    if (this.negate) stb.insert(0, '-');
    return correct(stb.toString());
    }

    private String correct(String str){
      StringBuffer stb=new StringBuffer(str);
      while ((stb.charAt(0)=='0')&&(stb.length()>1)){
        stb.deleteCharAt(0);
      }
      return stb.toString();
    }
    public boolean equals(KattaButunSon kbs){
      boolean res = false;
      if (this.compareTo(kbs)==0)
        res = true;
      else
        res = false;
      return res;
    }
    public KattaButunSon remainder(KattaButunSon kbs){
      String res = remainder(this.toString(), kbs.toString());
      return KattaButunSon.valueOf(res);
    }
    public String remainder(String bolin, String boluv){
      String[] res = divide(bolin, boluv);
      return res[1];
    }
    /*if first is less then second method returns - "0" */
    public KattaButunSon divide(KattaButunSon kbs){
      String[] divRes = divide(this.toString(), kbs.toString());
      return KattaButunSon.valueOf(divRes[0]);
    }
//  public String[] divide(KattaButunSon bolin,
//      KattaButunSon boluv){
//    return divide(bolin.toString(), boluv.toString());
//  }
    /*first[0]=integer div result
     *second[1]=remainder*/
    public String[] divide(String bolin, String boluv){
```

```
int addedIndex=0;
String[] result = new String[2];
if (boluv.equals("0")){
  System.out.println("Division by zero in KattaButunSon");
  result[0] = "0";
  result[1] = "0";
} else if (bolin.equals("0")){
  result[0] = "0";
  result[1] = "0";
} else if (boluv.equals("1")){
  result[0] = bolin;
  result[1] = "0";
} else {
  StringBuffer bolinma = new StringBuffer("");
  StringBuffer qoldiq = new StringBuffer("");
  KattaButunSon s1=new KattaButunSon(bolin);
  //System.out.println("\n\n"+compare(s1.toString(), s2.toString())+"\n\n\n");

  /*String[] str = getDivRes("91", "15");
  System.out.println("qol = "+str[0]);
  System.out.println("bo'lin = "+str[1]);*/

  int s1L=s1.length, s2L=boluv.length();
//  System.out.println("bolin = "+bolin+
//                     "   boluv = "+boluv);
//  System.out.println("s1L = "+s1L+"   s2L = "+s2L);
  if (compareTo(bolin, boluv)==-1){
        bolinma = new StringBuffer("0");
        qoldiq = new StringBuffer(bolin);
  } else if (compareTo(bolin, boluv)==0){
        bolinma=new StringBuffer("1");
        qoldiq = new StringBuffer("0");
  } else {
        //boolean bolindi = false;
        String[] forResult = new String[2];
        StringBuffer son1 = new StringBuffer(
              s1.toString());
        String son2 = boluv;//kbs.toString();
        StringBuffer tempStB = new StringBuffer();
        int curSonIndex=0, curNatija=0;
        int index=s2L-1;
        addedIndex=0;
        tempStB = new StringBuffer(
              son1.substring(0, s2L));
        //System.out.println("tempStB = "+tempStB);

        do {
              if (tempStB.charAt(0)=='0'){
                    tempStB.deleteCharAt(0);
              }
              //System.out.println("tempStB = "+tempStB);
              forResult = getDivRes(tempStB.toString(), son2);
```

```
                        //System.out.println("forResult[1] = "+forResult[0]);
                        bolinma.append(forResult[1]);

                        tempStB = new StringBuffer(forResult[0]);
                        addedIndex=0;
                        while (compareTo(tempStB.toString(), son2)<0){
                                index++;
                                //System.out.println("tempStB = "+tempStB);
                                if (index>s1L-1) break; else {
                                        addedIndex++;
                                        tempStB.append(son1.charAt(index));
                                }
                                if (addedIndex>1) bolinma.append('0');
                        }
                } while (compareTo(tempStB.toString(), son2)>-1);
                /*just try 18603/15=1240*/
                if (addedIndex==1) bolinma.append('0');
                qoldiq = tempStB;
        }
        if (addedIndex>1) bolinma.append('0');
        result[0] = correct(bolinma.toString());
        result[1] = correct(qoldiq.toString());
      }
      return result;
    }
    /*first[0]=integer div result
     *second[1]=remainder*/
    private String[] getDivRes(String s1, String s2){
      //System.out.println("s1 = "+s1+"\ns2 = "+s2);
      String result[] = new String[2];
      String temp = "";
      int res=0; String ayirma=s1;
      boolean flag = false;
      while (!flag){
        temp = ayirma;
        ayirma = remain(ayirma, s2);
        //System.out.println("ayirma = "+ayirma);
        if (ayirma.equals("-1")){
                flag =  true;
        } else {
                res++;
        }
      }
      result[0] = temp;
      result[1] = String.valueOf(res);
      return result;
    }

    /*Returns: -1, 0 or 1 as this KattaButunSon is numerically
     *less than, equal to, or greater than kbs.*/
    public int compareTo(KattaButunSon kbs){
      return compareTo(this.toString(), kbs.toString());
```

```
}
public static int compareTo(String s1, String s2){
  //System.out.println("compare("+s1+", "+s2+")");
  int res=-1, myInt=0;;
  int s1L=s1.length(), s2L=s2.length();
  if (s1L<s2L){ res=-1; } else {
    if (s1L==s2L){
        int length=s1.length();
        if ((s1L==s2L)&&(s1L!=1)){
            while (s1.charAt(myInt)==s2.charAt(myInt)){
                myInt++;
                if (myInt==length-1){
                    //System.out.println("OK 2");
                    break;
                }
            }
        }
        if ((myInt<s1L)&&(s1.charAt(myInt)<s2.charAt(myInt))){
            res=-1;
        }
        if ((myInt<s1L)&&(s1.charAt(myInt)>s2.charAt(myInt))){
            res = 1;
        }
        if ((myInt<s1L)&&(s1.charAt(myInt)==s2.charAt(myInt))){
            res = 0;
        }
    } else {
        res = 1;
    }
  }
  return res;
}

private String remain(String s1, String s2){
  KattaButunSon son1 = new KattaButunSon(s1);
  return (son1.remain(KattaButunSon.valueOf(s2))).toString();
}
public KattaButunSon remain(KattaButunSon s2){
  KattaButunSon s1=this;
  int s1L=s1.length, s2L=s2.length;
  int myInt = 0; boolean flag = false;
  byte[] son1, son2;
  //String result = "";
  StringBuffer stBRes = new StringBuffer("");
  /*if (s1.compareTo(s2)==-1){
    stBRes = new StringBuffer("-1");
  } else*/
  if (s1.compareTo(s2)==-1){
    stBRes = new StringBuffer("-1");
  } else {
    if (s1.compareTo(s2)==0){
        stBRes=new StringBuffer("0");
```

```
} else {
        if (s1L==s2L){
                while (s1.toString().charAt(myInt)==s2.toString().charAt(myInt)){
                        myInt++;
                        if (myInt==s1L-1) break;
                }
                if ((myInt<s1L)&&
                        (s1.toString().charAt(myInt)<s2.toString().charAt(myInt))){
                        stBRes = new StringBuffer("-1");
                        flag = true;
                }
        }
        if (!flag) {
                son1 = s1.byteView;  son2 = s2.getBytes();
                int index=0, notZeroIndex=0, temp=0;
                son1 = reverse(son1);
                son2 = reverse(son2);
                /*for (byte b: son1)
                        System.out.print(b+" ");
                System.out.println();*/

                while (index<s2L){
                        if (son1[index]-son2[index]>=0){
                                temp=son1[index]-son2[index];
                        } else {
                                notZeroIndex=index+1;
                                while (son1[notZeroIndex]==0){
                                        notZeroIndex++;
                                }
                                son1[notZeroIndex]-=1;
                                notZeroIndex--;
                                while (notZeroIndex!=index){
                                        son1[notZeroIndex]=9;
                                        notZeroIndex--;
                                }
                                son1[index]+=10;
                                temp=son1[index]-son2[index];
                        }
                        //result+=String.valueOf(temp);
                        stBRes.append(temp);
                        index++;
                }
                while (index!=s1L){
                        //result+=String.valueOf(son1[index]);
                        stBRes.append(son1[index]);
                        index++;
                }
                int myZeros = stBRes.length()-1;
                while ((stBRes.charAt(myZeros)=='0')&&(
                                        stBRes.length()>1)) {
                        stBRes.deleteCharAt(myZeros);
                        myZeros--;
```

```
                    }
                    stBRes = stBRes.reverse();
                }
            }
        }
        return KattaButunSon.valueOf(stBRes.toString());
    }
    private byte[] reverse(byte[] val){
        int lngth = val.length;
        byte[] res = new byte[lngth];
        for (int i=0; i<lngth; i++)
            res[i]=val[lngth-i-1];
        return res;
    }
    public KattaButunSon add(int a){
        return this.add(new KattaButunSon(
            String.valueOf(a)));
    }
    public KattaButunSon add(long a){
        return this.add(new KattaButunSon(
            String.valueOf(a)));
    }
    public KattaButunSon add(KattaButunSon b){
        int bigger = (length > b.length)? length: b.length;
        byte[] revA = new byte[length];
        byte[] revB = new byte[b.length];
        byte[] realB = b.getBytes();
        byte[] val = this.getBytes();

        byte[] resArr = new byte[bigger+1];
        /*2 ta n xonali son yig'indisi ko'pi bilan n+1 xonali bo'lishi mumkin**/

        for (int i=0; i<length; i++)
            revA[i] = val[length-i-1];

        for (int i=0; i<b.length; i++)
            revB[i] = realB[b.length-i-1];

        int temp=0, esda=0;
        for (int i=0; i<bigger; i++){
            if ((i<length)&&(i<b.length)){
                temp = (byte)(revA[i]+revB[i]+esda);
                resArr[i] = (byte)(temp%10);
                esda = temp/10;
            } else {
                if (i<length){
                    temp = (byte)(revA[i]+esda);
                    resArr[i] = (byte)(temp%10);
                    esda = temp/10;
                }
                if (i<b.length){
                    temp = (byte)(revB[i]+esda);
```

```
                    resArr[i] = (byte)(temp%10);
                    esda = temp/10;
                }
            }
        }
        if (esda != 0) resArr[bigger] = (byte)esda;

        StringBuffer stb = new StringBuffer();
    for (int i=0; i<resArr.length; i++)
        stb = stb.append(String.valueOf(resArr[i]));

    if (stb.charAt(stb.length()-1)=='0')
        stb = stb.deleteCharAt(stb.length()-1);
    stb = stb.reverse();

    KattaButunSon result = new KattaButunSon(new String(stb));
        return result;
    }

    public KattaButunSon multiply(KattaButunSon b){
        StringBuffer natija = new StringBuffer("");
        KattaButunSon result = new KattaButunSon("0");
        if ((this.equals(KattaButunSon.ZERO))||(
                b.equals(KattaButunSon.ZERO))){
            natija.append('0');
        } else {
            byte[] val = this.getBytes();
            byte[] bigger;// = (b.length>length)? b.getBytes(): val;
            byte[] smaller;// = (b.length>length)? val: b.getBytes();
            if (b.length>this.length){
                bigger = b.getBytes();
                smaller = val;
            } else {
                bigger = val;
                smaller = b.getBytes();
            }

            int bLength = bigger.length;
            int sLength = smaller.length;

            byte[] revB = new byte[bLength];
            byte[] revS = new byte[sLength];
            byte[] tempAdd = new byte[bLength+1];
            byte[] tempBigAdd = new byte[bLength+sLength+1];


            for (int i=0; i<bLength; i++)
                revB[i] = bigger[bLength-i-1];
            for (int i=0; i<sLength; i++)
                revS[i] = smaller[sLength-i-1];

    //      System.out.print("\nrevS--> ");
```

```
//   for (byte bt: revS)
//                   System.out.print(bt+", ");

     byte temp=0, esda=0, myB=0;
     KattaButunSon tempKattaButunSon;
     int tLength=0;

     for (int i=0; i<sLength; i++){
          esda = 0;
          for (int j=0; j<bLength; j++){
               temp = (byte)(esda + revS[i]*revB[j]);
               tempAdd[j] = (byte)(temp%10);
               esda = (byte)(temp/10);
          }
          if (esda!=0) tempAdd[bLength]=esda;
          tLength = tempAdd.length;

          StringBuffer stB = new StringBuffer("");
          for (int k=0; k<tLength; k++){
               stB.append(tempAdd[k]);
          }
          stB.reverse();
          if (stB.charAt(0)=='0') stB.deleteCharAt(0);
          for (int k=0; k<i; k++)
               stB.append('0');
//        System.out.println("stB = "+stB);

          tempKattaButunSon=new KattaButunSon(stB.toString());
          result = result.add(tempKattaButunSon);

          for (int j=0; j<tempAdd.length; j++){
               tempAdd[j] = 0;
          }
          for (int j=0; j<tempBigAdd.length; j++){
               tempBigAdd[j] = 0;
          }
          //System.gc();
     }
     natija.append(result.toString());
     while ((natija.charAt(0)=='0')&&(
          natija.length()>1)) natija.deleteCharAt(0);
   }
   return new KattaButunSon(natija.toString());
}
public String toBinaryString(){
   final String TWO = "2";
   StringBuffer stb = new StringBuffer("");
   KattaButunSon kbs = this;
   if (kbs.equals(KattaButunSon.ZERO)){
     stb.append('0');
   } else if (kbs.equals(KattaButunSon.ONE)){
     stb.append('1');
```

```
                } else {
                  String bolin = kbs.toString();
                  String[] bolmaVaQoldiq = new String[2];
                  boolean flag = true;
                  while(!((bolin.equals("0"))||(bolin.equals("1")))) {
                        bolmaVaQoldiq = divide(bolin, TWO);
                        bolin = bolmaVaQoldiq[0];
//                      System.out.println("bolin = "+bolin);
                        stb.append(bolmaVaQoldiq[1]);
//                      System.out.println("qoldiq = "+bolmaVaQoldiq[1]);
                  }
                  stb.append(bolin);
                  stb.reverse();
                }
                return stb.toString();
              }
}
```

web.xml of web application

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instanse/"
 xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
 version="2.4">

 <display-name>Confidential Information e-Document Exchange</display-name>
 <description>!!Final Work!!</description>

<servlet>
        <servlet-name>
                my_checkValidity
        </servlet-name>
        <servlet-class>

        nodir.web.edoc.CheckSignValidity
        </servlet-class>
 </servlet>
 <servlet-mapping>
        <servlet-name>
                my_checkValidity
        </servlet-name>
        <url-pattern>
                /checkValidity
        </url-pattern>
 </servlet-mapping>
<servlet>
        <servlet-name>
                my_admin_message_send
        </servlet-name>
        <servlet-class>
                nodir.web.edoc.FileSendAdmin
        </servlet-class>
 </servlet>
 <servlet-mapping>
        <servlet-name>
                my_admin_message_send
        </servlet-name>
        <url-pattern>
                /admin_message_send
        </url-pattern>
 </servlet-mapping>
<servlet>
        <servlet-name>
                my_admin_new_messages
        </servlet-name>
        <servlet-class>

        nodir.web.edoc.AdminNewMessages
        </servlet-class>
 </servlet>
 <servlet-mapping>
        <servlet-name>
                my_admin_new_messages
        </servlet-name>
        <url-pattern>
                /admin_new_messages
```

```
        </url-pattern>
    </servlet-mapping>
    <servlet>
        <servlet-name>
                my_message_detail
        </servlet-name>
        <servlet-class>

        nodir.web.edoc.TakeMessageDetail
        </servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>
                my_message_detail
        </servlet-name>
        <url-pattern>
                /message_detail
        </url-pattern>
    </servlet-mapping>
    <servlet>
        <servlet-name>
                checkuserlogin
        </servlet-name>
        <servlet-class>

        nodir.web.edoc.CheckUserLogin
        </servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>
                checkuserlogin
        </servlet-name>
        <url-pattern>
                /checkuserlogin
        </url-pattern>
    </servlet-mapping>
    <servlet>
        <servlet-name>
                mysendname
        </servlet-name>
        <servlet-class>
                nodir.web.edoc.FileSend
        </servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>
                mysendname
        </servlet-name>
        <url-pattern>
                /mysend
        </url-pattern>
    </servlet-mapping>
    <servlet>
        <servlet-name>
                upload
        </servlet-name>
        <servlet-class>

        nodir.web.edoc.FileUploadServlet
        </servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>
                upload
        </servlet-name>
        <url-pattern>
                /myupload
        </url-pattern>
    </servlet-mapping>
    <servlet>
        <servlet-name>
                check_admin_name
        </servlet-name>
        <servlet-class>
                nodir.web.edoc.CheckAdmin
        </servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>
                check_admin_name
        </servlet-name>
        <url-pattern>
                /check_admin
        </url-pattern>
    </servlet-mapping>

        <servlet>
        <servlet-name>
                reguser
        </servlet-name>
        <servlet-class>
                nodir.web.edoc.RegUser
        </servlet-class>
        </servlet>
    <servlet-mapping>
        <servlet-name>
                reguser
        </servlet-name>
        <url-pattern>
                /regnewuser
        </url-pattern>
    </servlet-mapping>

    </web-app>
```

ЎЗБЕКИСТОН РЕСПУБЛИКАСИ

ДАВЛАТ ПАТЕНТ ИДОРАСИ

ЭЛЕКТРОН ҲИСОБЛАШ МАШИНАЛАРИ УЧУН ЯРАТИЛГАН ДАСТУРНИНГ РАСМИЙ РЎЙХАТДАН ЎТКАЗИЛГАНЛИГИ ТЎҒРИСИДАГИ

ГУВОҲНОМА

№ DGU 01358

Ушбу гувоҳнома Давлат патент идораси томонидан Ўзбекистон Республикасининг 1994 йил 6 майда қабул қилинган «Электрон ҳисоблаш машиналари учун яратилган дастурлар ва маълумотлар базаларининг ҳуқуқий ҳимояси тўғрисида»ги Қонунига асосан

Ёпиқ сандиқ алгоритми

номли электрон ҳисоблаш машиналари учун яратилган дастурга берилди.

23.08.2007 йилда келиб тушган **№ DGU 2007 0139** талабнома бўйича

Ҳуқуқ эгаси(лари): ***Хамдамов Рустам Хамдамович, Кодиров Нодир Хомиджонович, UZ***

Муаллиф(лар): ***Хамдамов Рустам Хамдамович, Кодиров Нодир Хомиджонович, UZ***

Ўзбекистон Республикаси электрон ҳисоблаш машиналари учун дастур давлат реестрида **31.08.2007** йилда Тошкент шаҳрида рўйхатдан ўтказилган.

Директор

А.А. Азимов