Research Ontology Data Models for Data and Metadata Exchange Repository

Iryna Kamenieva
Master Thesis

Research Ontology Data Models for Data and Metadata Exchange Repository

Iryna Kamenieva

Supervisor: Dr. Marcelo Milrad
Abstract

For researches in the field of the data mining and machine learning the necessary condition is an availability of various input data set. Now researchers create the databases of such sets. Examples of the following systems are: The UCI Machine Learning Repository, Data Envelopment Analysis Dataset Repository, XMLData Repository, Frequent Itemset Mining Dataset Repository. Along with above specified statistical repositories, the whole pleiad from simple filestores to specialized repositories can be used by researchers during solution of applied tasks, researches of own algorithms and scientific problems. It would seem, a single complexity for the user will be search and direct understanding of structure of so separated storages of the information. However detailed research of such repositories leads us to comprehension of deeper problems existing in usage of data. In particular a complete mismatch and rigidity of data files structure with SDMX - Statistical Data and Metadata Exchange - standard and structure used by many European organizations, impossibility of preliminary data origination to the concrete applied task, lack of data usage history for those or other scientific and applied tasks.

Now there are lots of methods of data miming, as well as quantities of data stored in various repositories. In repositories there are no methods of DM (data miming) and moreover, methods are not linked to application areas. An essential problem is subject domain link (problem domain), methods of DM and datasets for an appropriate method. Therefore in this work we consider the building problem of ontological models of DM methods, interaction description of methods of data corresponding to them from repositories and intelligent agents allowing the statistical repository user to choose the appropriate method and data corresponding to the solved task. In this work the system structure is offered, the intelligent search agent on ontological model of DM methods considering the personal inquiries of the user is realized.

For implementation of an intelligent data and metadata exchange repository the agent oriented approach has been selected. The model uses the service oriented architecture. Here is used the cross platform programming language Java, multi-agent platform Jadex, database server Oracle Spatial 10g, and also the development environment for ontological models - Protégé Version 3.4.

Keywords: repository, SDMX standart, data mining, classification, textual collection,hierarchical data model, semantic web, ontology, multiagent system, search algorithms, agent-oriented systems, intelligent agent, jadex, sdk, java, rdf, protégé, sparql, oracle sprotiat.
Acknowledgements

I would like to express my gratitude to my supervisor Dr. Marcelo Milrad, who expressed interest in my work, encouraged, stimulated and helped me with this thesis. I am thankful to Tatyana Shatovska and Victoriya Repka, who gave me strong and initial ideas about the work and especially for encouraging and supporting my efforts connected with the thesis. I also thank my family for supporting me morally all the time, they were always with me, and I thank all my friends for spending nice times with me, even during hard working days.
Table of content

1. INTRODUCTION ......................................................................................................................... 1
   1.1  PROBLEM DEFINITION .............................................................................................................. 1
   1.2  GOALS AND CRITERIA .............................................................................................................. 2
   1.3  PURPOSE OF WORK .................................................................................................................. 3
   1.4  LIMITATIONS ........................................................................................................................ 3
   1.5  OUTLINE OF THIS THESIS ........................................................................................................ 3

2. METHODOLOGICAL APPROACH ................................................................................................. 4
   2.1  SOFTWARE DEVELOPMENT APPROACH .................................................................................. 4
   2.2  THE PROCESS OF IMPLEMENTATION DEVELOPMENT .......................................................... 4

3. DATA MIMING REPOSITORIES RESEARCH ................................................................................. 7
   3.1  UCI MACHINE LEARNING REPOSITORY ................................................................................. 7
   3.2  DEA DATASET REPOSITORY .................................................................................................. 8
   3.3  XML DATA REPOSITORY ........................................................................................................ 9
   3.4  FREQUENT ITEMSET MINING DATASET REPOSITORY ........................................................... 10
   3.5  ANALYTICAL RESEARCH ...................................................................................................... 11
   3.6  ONTOLOGY DATA AND METADATA EXCHANGE REPOSITORY ........................................... 11

4. MULTI-AGENT INTELLECTUAL TECHNOLOGIES ....................................................................... 12
   4.1  AGENT TECHNOLOGY ............................................................................................................. 12
   4.2  BASIC CONCEPTS OF AGENT APPROACH .............................................................................. 13
   4.3  INTELLIGENT AGENT’S FEATURES ....................................................................................... 13
   4.4  BELIEF-DESIRE- INTENTION (BDI) AGENT ARCHITECTURE .................................................. 13
   4.5  MULTI-AGENT SYSTEM (MAS) ................................................................................................ 14
   4.6  MAS CLASSIFICATION ............................................................................................................. 14
   4.7  AGENT ENVIRONMENT .......................................................................................................... 15
   4.8  AGENT-ORIENTED SYSTEM .................................................................................................... 15

5. ONTOLOGY MODELS DEVELOPMENT ......................................................................................... 21
   5.1  IMPLEMENTATION .................................................................................................................... 21
   5.2  ONTOLOGY REPRESENTATION ............................................................................................... 21
   5.3  PROGRAM - INSTRUMENTAL METHOD OF IMPLEMENTATION OF THE ONTOLOGICAL MODEL ................................................................. 22
   5.4  ONTOLOGY SOURCE MODEL (DATASET ONTOLOGY MODEL) ............................................ 22
      5.4.1  Ontological source models development ........................................................................ 22
      5.4.2  The ontological models source of classes description ..................................................... 24
   5.5  ONTOLOGY DATA MINING MODEL ........................................................................................ 25
      5.5.1  Ontological data mining models development ................................................................. 25
   5.6  ONTOLOGY USER MODEL ..................................................................................................... 27
      5.6.1  RDF model ...................................................................................................................... 27
      5.6.2  Ontological user models development ............................................................................ 28
      5.6.3  The ontological user models of classes description ....................................................... 29

6. INTELLIGENT SEARCH AGENT DESIGN AND DEVELOPMENT ............................................. 31
   6.1  AGENT IMPLEMENTATION .................................................................................................... 31
   6.2  INTELLIGENT SEARCH AGENT ............................................................................................. 31
   6.3  THE SEARCH AGENT GOALS ................................................................................................ 33
   6.4  SEARCH AGENT OUTLINE .................................................................................................... 35
   6.5  SEARCH AGENT ADF ............................................................................................................ 36
   6.6  SEARCH AGENT “BELIEF” .................................................................................................... 36
   6.7  SEARCH AGENT INTERACTION WITH OTHER AGENTS ...................................................... 37
      6.7.1  Search agent Scenario ...................................................................................................... 37
      6.7.2  User agent Scenario ........................................................................................................ 38
      6.7.3  Coordinator [Manager] agent Scenario ........................................................................... 43
      6.7.4  Source agent Scenario .................................................................................................... 46
      6.7.5  Classification Scenario .................................................................................................... 49
   6.8  AGENTS DEVELOPMENT USING JADEX TECHNOLOGY .................................................... 49

7. SYSTEM PROGRAM MODEL: DEPLOYMENT AND IMPLEMENTATION .................................. 50
   7.1  PROBLEM-SOLVING .............................................................................................................. 50
7.2 PRESENTATION LEVEL.........................................................................................................................51
7.3 SERVICE LEVEL.................................................................................................................................54
7.4 AGENT SUBSYSTEM..........................................................................................................................55
7.5 DATA AND METADATA EXCHANGE REPOSITORY EXPLANATION ..................................................55
7.6 WORK DATABASE LEVEL ..................................................................................................................56

8. CONCLUSIONS AND FUTURE CHALLENGES..................................................................................58
  8.1 RESULTS ...............................................................................................................................................58
  8.2 CONCLUSIONS .....................................................................................................................................59
  8.3 FUTURE CHALLENGES .....................................................................................................................59
  8.4 REFLECTIONS .......................................................................................................................................60

REFERENCES..............................................................................................................................................62

INTERNET SITES ......................................................................................................................................65

APPENDICES ...........................................................................................................................................66
  APPENDIX A DATA AND METADATA EXCHANGE REPOSITORY (DATA MINING REPOSITORY) [AN EXAMPLE] ..........66
  APPENDIX B DATA AND METADATA EXCHANGE REPOSITORY (DATA MINING REPOSITORY) .............................69
List of Figure

FIGURE 2.1: SDMX INFORMATION MODEL ................................................................. 5
FIGURE 2.2: METADATA SCHEME DESCRIPTION .................................................. 5
FIGURE 3.1: UCI REPOSITORY WEB-PAGE ............................................................ 7
FIGURE 3.2: HOME PAGE DEA ................................................................. 8
FIGURE 3.3: DEA PAGE LOGIN ........................................................................... 8
FIGURE 3.4: XMLDATA REPOSITORY INTERFACE ........................................... 9
FIGURE 3.5: DATASET INFORMATION ............................................................ 9
FIGURE 3.6: FIMI DATASETS ............................................................................ 10
FIGURE 4.1: DELIBERATIVE ARCHITECTURE BASE ....................................... 16
FIGURE 4.2: REACTIVE ARCHITECTURE BASE ............................................... 17
FIGURE 4.3: HYBRID MULTI-LAYER AGENT BASE ARCHITECTURE .............. 19
FIGURE 4.4: AGENT STRUCTURE OF CYCLIC-MACHINE ARCHITECTURE ........ 20
FIGURE 5.1: ONTOLOGY SOURCE MODEL CLASSES AND ATTRIBUTES IN THE PROTEGÉ-3.4 ............................................ 23
FIGURE 5.2: DATA MINING ONTOLOGY, WHERE ........................................... 26
FIGURE 5.3 DATA MINING ONTOLOGY MODEL ............................................... 27
FIGURE 5.4: PART OF RDF MODEL ............................................................... 28
FIGURE 5.5: USER ONTOLOGY MODEL ......................................................... 29
FIGURE 6.1: THE USE CASE DIAGRAM IN TERMS OF USER ................................ 32
FIGURE 6.2: THE USE CASE DIAGRAM IN TERMS OF INTERACTION BETWEEN AGENTS .............................................................................. 32
FIGURE 6.3: SEARCH AGENT MAIN GOALS .................................................. 33
FIGURE 6.4: THE DIAGRAM OF SIMPLE SEARCH GOALS ............................... 34
FIGURE 6.5: THE SCHEME OF ADVANCED SEARCH GOALS ............................ 34
FIGURE 6.6: THE STRUCTURE OF THE SEARCH AGENT PLANS ...................... 35
FIGURE 6.7: THE SEARCH AGENT GOALS DESCRIPTION .................................. 36
FIGURE 6.8: SEARCH AGENT STRUCTURE ..................................................... 38
FIGURE 6.9: THE USE CASE DIAGRAM FOR USER AGENT .............................. 39
FIGURE 6.10: THE USER AGENT CLASS DIAGRAM ......................................... 42
FIGURE 6.11: THE USE CASE DIAGRAM FOR COORDINATOR AGENT .......... 43
FIGURE 6.12: THE DIAGRAM OF MANAGER AGENT CLASSES ....................... 45
FIGURE 6.13: THE SEQUENCES DIAGRAM - NEW DATASET ADDING TO THE REPOSITORY ................................................................. 47
FIGURE 6.14: THE SEQUENCES DIAGRAM REVIEW OF ALL REPOSITORY DATASETS .................................................................................................................. 47
FIGURE 6.15: THE DIAGRAM OF SEQUENCES DATASET ASSESSMENT ............ 48
FIGURE 6.16: THE DIAGRAM OF SOURCE AGENT CLASSES ........................... 48
FIGURE 7.1: THE OVERALL STRUCTURE OF THE SYSTEM ................................ 51
FIGURE 7.2: WICKET PAGES WORK SCHEME ............................................... 51
FIGURE 7.3: MARKUP CREATEDATASETPAGE ............................................... 52
FIGURE 7.4: THE USE CASE DIAGRAM ........................................................... 52
FIGURE 7.5: THE PRESENTATION LEVEL AND SERVICE LEVEL DIAGRAM CLASS .................................................................................. 54
FIGURE 7.6: THE EXPLANATION OF OVERALL SYSTEM ................................ 545
FIGURE 7.7: A MODEL TO STORE RDF STATEMENTS IN ORACLE SPATIAL 10G ................................................................. 56
FIGURE A.1: THE LOG AND PASSWORD PAGE ................................................. 66
FIGURE A.2: BEGINNER REGISTRATION .......................................................... 66
FIGURE A.3: ADVANCED USER REGISTRATION .............................................. 67
FIGURE A.4: RESEARCH METHODS INFORMATION ........................................ 67
FIGURE A.5: VIEW USER INFORMATION .......................................................... 68
FIGURE A.6: THE USER REFRESHMENT ............................................................ 68
FIGURE B.1: SIMPLE SEARCH PAGE .............................................................. 69
FIGURE B.2: ADVANCED SEARCH PAGE ........................................................ 69
FIGURE B.3: SEARCH RESULTS PAGE ............................................................ 70
FIGURE B.4: THE PROMPT-PAGE OF THE SYSTEM OF THE POPULAR QUERIES ................................................................. 70
FIGURE B.5: THE LIST OF THE MOST POPULAR DATASETS OF REPOSITORY ON THE HOME PAGE ................................................................. 71
List of Tables

TABLE 5.1: DATASET CLASS SLOTS ......................................................................................................................... 24
TABLE 5.2: DATASETFILE CLASS SLOTS .................................................................................................................. 24
TABLE 5.3: JUDGE CLASS SLOTS ............................................................................................................................. 25
TABLE 5.4: ADDRESS SLOTS CLASS ........................................................................................................................ 29
TABLE 5.5: UNIVERSITY SLOTS CLASS .................................................................................................................. 29
TABLE 5.6: PREFERENCE SLOTS CLASS .................................................................................................................. 30
TABLE 5.7: SLOTS OF ABSTRACT CLASS ACCOUNT ............................................................................................ 30
TABLE 5.8: SLOTS OF ABSTRACT CLASS PERSON ................................................................................................. 30
TABLE 5.9: SLOTS OF EXPERIENCED CLASS ....................................................................................................... 30
TABLE 6.1: THE SEARCH AGENT KNOWLEDGE DESCRIPTION ............................................................................... 37
TABLE 6.2: THE SEARCH AGENT EVENTS ............................................................................................................. 37
TABLE 6.3: THE USER AGENT KNOWLEDGE ....................................................................................................... 40
TABLE 6.4: THE AGENT USER EVENTS ................................................................................................................ 41
TABLE 6.5: COORDINATOR AGENT KNOWLEDGE .............................................................................................. 44
TABLE 6.6: THE COORDINATOR AGENT EVENTS ............................................................................................... 45
1 Introduction

We envisage a world where the barriers to sharing and exchanging data and information are radically lowered. The world we are undoubtedly moving toward is one of Web-based ‘mash-ups’; that is, networked software applications that can combine data in real-time from multiple service providers in ways that are user-friendly, yet powerful. To be effective in this space, it is imperative that repositories become first-class service providers. Repository is a dataset of statistical data, which contain data, data description, and metadata description to them. This involves collecting, curating and preserving good metadata. Hence, metadata is not simply about technical requirements specific to repositories; rather, it forms the basis of an emerging information infrastructure for data stores communications that has far-reaching consequences.

Digital repositories are networked software applications primarily used for storing, managing and disseminating data (e.g. digital publications, theses, data sets and so on). The Repositories differ from conventional content management systems because they include technologies to ensure that data are preserved for long-term access and use. Although repositories were initially developed for science purpose, statistical companies, they are currently being implemented more widely; for example, by museums to facilitate online access to cultural heritage resources, and government agencies to mediate long-term access to documents and other data. In practical terms, implementing a digital repository nowadays can be as simple as downloading free open-source software and installing it onto a networked computer.

Establishing a stable repository for everyday institutional use is an altogether harder proposition however. The most popular open source repository applications are DSpace, Fedora, E-prints, UCI Knowledge Discovery in Databases Archive; DEA DATASET REPOSITORY; Frequent Itemset Mining Dataset Repository; XMLData Repository. There are some commercial repository software providers, but none have gained the same level of popularity as the open source repositories mentioned. The important point to note here is that a repository is essentially a relational database that stores and keeps track of metadata records for files stored in a mass-data storage facility.

The underlying technology is relatively straightforward whereas the institutional context of use is typically complex. These systems are not information. It is difficult to exchange files automatically. In the next chapters I will describe in greater detail how to create intellectual search agent for public information repository based on ontologies models and intellectual agents.

1.1 Problem Definition

In this work the conceptual structure and interaction principles of intelligent agents and ontological models in the intelligent data and metadata exchange repository will be offered. The main attention in this work will be paid to the development of intelligent search agent model realizing information extraction on ontological model of DM methods. Parts of ontological models of DM regarding clusterization, classifications, predictions with usage of query language SPARQL. In a client part of system there is considered the building of the intelligent agent of the repository user, the coordinator (manager) agent, which controls the common state of the system, and also fulfils the registration and authorizations of users, the resource (dataset) agent with partial usage of files structure with SDMX standard data.

The RDF language was used to describe and to develop the ontological model of the user, ontological model of DM, ontological model of resources. Choosing the development methods the service- and agent oriented technologies have been combined into the uniform architecture. The three-level architecture has been selected for implementation of this system, which main body of business logic was built by means of agent technology Jadex, allowing,
following Belief-Desire-Intention (BDI) model, to develop the goal agents. Jadex includes this model in Java Agent Development Framework (Jade) agents, adding representations, goals and schedules, as objects of the first class, which can be created and used within the agent.

Jade is the software environment intended for development of the multi-agents systems and the applications, supporting FIPA-standards for intelligent agents (Weiss et al., 1997). At data representation level there is a Web application. The Web services are as a part of business logic and a layer between representation level and business logic and data storage levels.

1.2 Goals and Criteria
Creating a new public information repository to store datasets using intelligent agent and ontology approach for storing, conversion, search, add, description, selection of the required information for researchers needs in the field of Data mining and Machine Learning. As a base standard was chosen the standard SDMX Standards Version 2.0 and the main parameters of the Statistical European Repositories were taken.

The main reason to create information statistical exchange repository is to improve structure of repositories using ontologies and intellectual agents.

The aim of this work is to study and develop an algorithm and architecture of the multi-agent system module searching, which provides the ontology models for data and metadata exchange repository (data mining repository). An analysis of subject area to identify further development in and to formulate approaches against search engines and their implementation based on agent technology, services and data models based on ontologies is necessary. To develop a prototype system it is necessary to:

- examine modern methods of finding statistical repositories;
- analyze intelligent agents, multi-systems, agent-oriented approach;
- develop a search algorithm to ontological models (simple and advanced search supporting, account search, which takes individual interests, orientation of activities, previous search queries of the user, to explore different strategies of search algorithm);
- research existing systems and platforms implementation of building systems based on intelligent agents;
- develop a model for integrating intelligent agents with web systems.
- develop a model of intelligent search agent, and its relationship to other agents.

The search agent should be able to:

- perform a simple search for users regardless of user type;
- search by different criteria for authorized users;
- provide popular data sets;
- perform a search taking into account the personal needs of the user;
- provide user relevant queries information;
- keep statistics of requests and, if necessary, provide this information;
- remember the successful search results.
- design and implement the architecture of search module (research capabilities of intelligent agents in the field of search and retrieval systems, the choice of intelligent agent type, the development of the architecture search module based on agent-oriented methods, the implementation of the developed architecture based on multi-platform for intelligent agents) based on search algorithm;
- test search module of ontology data model for data and metadata exchange repository on a large amount of input data;
- research and analyze various strategies for the search module (analysis and testing various strategies to search engine, search optimization).
1.3 Purpose of work
The conceptual idea is to create a structure for intelligent information data exchange system.
We will focus on the development of the ontology model for data mining methods, ontology model for data transformation methods and intellectual search agent for collaboration between these models, datasets and user profile. In our work we will use SDMX standard (SDMX, 2005) for dataset description.
Text mining methods will be implemented as part of the classification problem for unlabeled datasets.
In this system the following will be fulfilled:
- search agent program realization
- ontology models
- “ontology data and metadata exchange repository” architecture
- text classification (clusterization) for datasets

1.4 Limitations
The “data and metadata exchange repository” is not entirely integrated with the rest of the statistics repository. The system is also not fully integrated with the areas of data mining and machine learning, for full and detailed description of these areas.

1.5 Outline of this thesis
Chapter 2 introduces the main idea behind this thesis, and the state of the art technologies by describing some important issues related to metadata, ontologies, intellectual agents, SDMX standard. It also gives a brief description of other related technologies to artificial intelligence. Chapter 3 reviews the possibilities of modern methods of searching for statistical repositories and provides insight to “data and metadata exchange repository”. Chapter 4 describes intellectual system architecture based on intellectual agents. The architecture is explained based on Service Oriented Architecture of a distributed system and multi-agent systems. Chapter 5 describes the development of ontology models and their technology creating. Chapter 6 presents a number of intellectual agent development, description and scenarios and text clusterization (classification) approach for our system. Each scenario describes the flow of events according to the agent’s activity. Text clusterization (classification) approach was created. Chapter 7 elicits the architecture of the “data and metadata exchange repository”. Chapter 8 summarizes the fundamental idea and the results that were obtained based on the work conducted in this thesis. The implementation of the application is also described shortly by illustrating some of the main developed parts concerning the architecture presented in Chapters 5, 6, 7.
2 Methodological approach

The basic idea of the intellectual repository system is to provide for the user a particular set of analytical datasets in accordance of his objectives, as well as the most effective Data Mining method for processing such a set. One of the most effective intellectual methods of formal description of any subject area is the ontological approach.

2.1 Software development approach

In this work we use a formal description of the Data Mining methods based on the formal language of the Resource Description Framework - RDF, which is the most unified for description any resources in the form of a directed graph. All the information will be saved as a set of ontological models. For the ontological models processing (search activities on the models), it is used a query language. We will use the query language SPARQL, which has been standardized by RDF working group.

To implement the query to the ontological models using the SPARQL language we have to use an intellectual model, which will join the user's query to the system and search methods on the models, as well as information input. For this purpose, in the system the strategy of the intelligent agents building will be used. Intelligent agents are a program entity which autonomously operates to achieve the agents or user goals and has the intellectual characteristics. To implement the multi-agents system we have used instrumental platform Jadex. For agent’s functionality description and for description of their interactions, as well as for description the sequence of requests to the system we have used a universal modeling language UML.

To implement multi-agents system using ontological representation and storage of the information we have developed client-server application architecture using Web services. The main elements of such system would be: Information in the form of ontological models, intelligent agents that will process the data, give necessary information to the user, user interface, which allows him to set a formal requests, and if it is necessary to refine requests to get more understandable results. As the database server we have used DB Oracle Spatial.

2.2 The process of implementation development

Let us consider some of the stages of the system implementation.

1. Requirement analysis

First of all repositories of scientific collections of statistical data, which identified their strengths and weaknesses were deeply researched and analyzed. This analysis has helped on the basis of the characteristics of the analysis of the shortcomings of existing repositories of scientific data sets to develop a software implementation. It solves the problem of preservation of large and stable sets of data using ontological models in the hierarchical structures and improves the efficiency of working with them.

2. Models design

The Ontology is a complete structural specification of a certain subject area, its formal submission, which includes a glossary of terms of that area and the set of logical relations, which describe how these terms relate to each other.

Ontologies allow creating an effective information exchange system. The main task is not to collect disparate information, but structured, formal data to solve real business and economic challenges. The main purpose of information exchange system is to make information accessible and reusable across the whole system. Due to this fact that information, which is not described and not structured, eventually becoming worthless. In contrast, information, which allows automated distribution and exchange generates added value. The entire above problem is solved in the system (Ratyshin et al., 2001).

The ontological models of intellectual processing of the data user, data sets, resources,
external systems for integration and sharing data were designed. It was necessary to develop a search algorithm (the agent) with the least loss of time, passing on the hierarchical structure of ontological models could qualitatively authenticated information.

As a basic standard description of the data set the SDMX Standards Version 2.0 (SDMX, 2005) was used so basic parameters of the statistical description of European repositories for automatic implementation of data integration between the repositories. The structure of the package is shown in Figure 2.1.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Metadata set</th>
<th>Data &amp; Metadata Provisioning</th>
<th>Subscription &amp; Notification</th>
<th>Registration</th>
<th>Discovery</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Family</td>
<td>Metadata Structure Definition</td>
<td>Concept Scheme</td>
<td>Category Scheme</td>
<td>Code List</td>
<td>Hierarchical Code Scheme</td>
</tr>
</tbody>
</table>

*Figure 2.1: SDMX information model*

These models using Protégé 3.4 were created. The Protégé 3.4 is an integrated tool based on knowledge for designers of systems. The Protégé 3.4 contains a model of knowledge, which consists of the classes of information, slots, instances and applications. The Protégé 3.4 tool can access all these parts with the help of the unified graphical user interface. The upper level includes overlapping tabs for compact presentation of the parts and their co-editing. This top-level design to the tabs allows the modeling integration of the ontology of classes describing a particular method, creating a means of learning to gather information, enter the individual items of data and building a knowledge base.

Metadata is data about data. Metadata does not have the information, but describes the attributes of data containing information (e.g., not the name of the customer, but that field «Customer Name» has a length of 35 characters, composed of uppercase and lowercase letters, and is linked to the field «name»). Metadata is stored in the form of database tables in a repository. Their maintenance is carried out centrally. Metadata purpose is to control the attribute data consistency in the system and facilitate data management by adjusting the attributes in one location. The results of the adjustment will be automatically distributed to all necessary applications. In order to integrate the repository with data warehouse was used as the basic scheme of the standard and their own specific concepts Figure 2.2.

*Figure 2.2: Metadata scheme description*
Interaction between the ontological models is based on intelligent agents: the agent coordinator, a resource agent, agent search, a user agent.

The creation of ontologies is a promising area of modern research in the processing of information provided in natural language. One of the advantages of using ontologies as a tool for learning is a systematic approach to the study of the subject area. This is achieved: regularity in ontology provides a holistic view of the subject area, uniformity in the material presented in a unified format is much better perceived and reproduced; scientific in building the ontology allows restoring the missing logical link in their entirety. Also, ontologies allow using large amounts of data from different systems due to the creation of semantic description of data.

3. Model and agent development

The Intelligent search agent was developed. The intelligent agent is some systems, which have the following characteristics:

- **Autonomy** - the action of agent is determined only by its internal state, no external stimuli can influence the behavior of the policy agent if it has not been foreseen by its structure;
- **Reactivity** - agents exist within a certain environment, which interact, i.e. able to perceive changes in the environment and respond to them;
- **Proactive** - agents have a goal-directed behavior to solve their problem, i.e. agent tries to solve the task entrusted to it in a changing environment, which is planning its own actions;
- **Social ability** - agents are able to interact and collaborate with each other for the task, interact with the ontological models to get good results;
- **Personal picture of the world**: each agent has its own model of the surrounding world (environment), which describes the manner in which the agent sees the world. The agent bases its model of peace on the basis of information received by the external environment;
- **Sociability and cooperativity**: Agents can exchange information with their environment and other agents. The possibility of communication means that the agent should receive information about its environment, which gives it the opportunity to build their own model of the world. Moreover, the possibility of communication with other agents is a prerequisite for joint action to achieve goals;
- **Intelligent behavior**: the behavior of an agent includes the ability to learn, logical deduction, or construction of a model environment in order to find the best ways to conduct.

Therefore, each agent is a process that has a certain part of knowledge about the object and the opportunity to share this knowledge with other agents.

As a result, intelligent search agent interacts with the ontological model and the user to get an expert answer to the query.
3 Data mining repositories Research

In this chapter we present analysis of the most popular statistical repositories, analytical comparative analysis about each of them and short overview of the system ontology data and metadata exchange repository.

The Repository is a place where any data are stored and maintained. The most common data repository is stored in files accessible to the further spread of the network (Pearson, et al., 2004; Cunningham, et al., 2008). The repository is a database of configuration management and change management throughout the life cycle of a data warehouse. Also the repository contains all the necessary information to interested individuals in the stages of its creation and operation (Xie, et al., 2006). The repository stores the basic version of the software data warehouse, data that reflect the history of its establishment and operation, detect errors claims during its operation and the requirements and wishes of its modernization and a complete set of documentation for the version of the software data warehouse (Zimmermann, 2006; Neil, 2005-2008). It also keeps detailed information on the processes of development and maintenance (Moore, et al., 2009; Fatudimu, et al., 2008).

3.1 UCI Machine Learning Repository

UCI Machine Learning Repository is the largest repository of real and model tasks of machine learning (Fig. 3.1). Repository containing real data on applications in biology, medicine, physics, engineering, sociology, and other data repository that is widely used by students, teachers and researchers around the world as the primary source of data for empirical analysis, testing and comparison of machine learning algorithms. UCI repository as ftp established at the University in the Irvin city (California, USA) (Blake C. L. et al., 2001), (Cortez, et al., 2007), (Asuncion A., et al., 2007).

![Figure 3.1: UCI Repository web-page](image-url)
Public repository Advantages:
- provided the opportunity to play and verification of results by other researchers;
- because many problems has aggravated «fitting» algorithm for one specific task;
- the algorithm is best to solve an opportunity to provide classes of tasks.

The advantages of this repository: well-sorted data, full text search.
Disadvantages: only text data is not easy to use and change, the lack of search.
This is one of the few repositories, which has a reputation as a reliable repository of scientific data sets. Deserves respect data sets filtering.

3.2 DEA Dataset Repository
Several DEA Dataset Repository were created using ASP.NET technology and databases. The main page of the DEA web-site contains information about DEA Datasets repository presented in Figure 3.2.

**WELCOME TO THE DEA DATASET REPOSITORY**
Over 500 members of the DEA research and practitioner community are already listed here. You can upload your DEA datasets information and files in our dataset library, or you can search our entire library of dataset and download any useful information you may required.

Please click on the button below to proceed.

![Figure 3.2: Home page DEA](image)

Clicking on the “Continue” button on the home page, you can go to the login. It is represented in Figure 3.3.

![Figure 3.3: DEA Page login](image)

The advantage of this system is convenient resources search.
3.3 XML Data Repository

In the XML Data Repository data is stored in XML format, as well as statistical data for use in research experiments. Interface and information about datasets is presented in Figure 3.4 and Figure 3.5. Large volumes of data is stored in compressed form using GZIP and XMILL. Statistics data set is calculated using the XML Toolkit.

XML Data Repository

**Goal**

The XML Data Repository collects publicly available datasets in XML form, and provides statistics on the datasets, for use in research experiments. Whenever possible, DTDs for the datasets are included and the datasets are validated. Some of the datasets are large, and each is provided in compressed form using gzip and XMILL. Dataset statistics were computed using the XML Toolkit.

**Detailed View of Datasets**

**Repository Contents**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Uncompressed Size</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protein Sequence Database</td>
<td>Integrated collection of functionally annotated protein sequences</td>
<td>603 MB</td>
<td>Nov 9 2001</td>
</tr>
<tr>
<td>ScorePost</td>
<td>SWISS-Model is a curated protein sequence database which strives to provide a high level of annotations (such as the description of the function of a protein, its domain structure, post-translational modifications, variants, etc.), a minimal level of redundancy and a high level of integration with other databases</td>
<td>169 MB</td>
<td>1998</td>
</tr>
<tr>
<td>Auction Data</td>
<td>Auction data converted to XML from web sources</td>
<td>22 KB</td>
<td>2001</td>
</tr>
<tr>
<td>DBLP Computer Science Bibliography</td>
<td>The DBLP server provides bibliographic information on computer science journals and proceedings. DBLP stands for Digital Bibliography Library Project</td>
<td>127 MB</td>
<td>Oct 2002</td>
</tr>
<tr>
<td>Temporary Oligos</td>
<td>Oligo data derived from university websites</td>
<td>277 KB</td>
<td>1999</td>
</tr>
</tbody>
</table>

**Figure 3.4: XML Data Repository Interface**

Administration of XML Data Repository engages just one person. To add data to the repository, you should send the information to the specified e-mail address.

**Datasets, Details, and Download**

**Protein Sequence Database**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Uncompressed Size</th>
<th>DTD</th>
<th>Elements</th>
<th>Attributes</th>
<th>Max Depth</th>
<th>Avg Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>proteinDB</td>
<td></td>
<td>XML (603 MB)</td>
<td>xsd</td>
<td>21300183</td>
<td>1298047</td>
<td>7</td>
<td>5.15147</td>
</tr>
</tbody>
</table>

**ScorePost**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Uncompressed Size</th>
<th>DTD</th>
<th>Elements</th>
<th>Attributes</th>
<th>Max Depth</th>
<th>Avg Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScorePost</td>
<td>SWISS-Model is a curated protein sequence database which strives to provide a high level of annotations (such as the description of the function of a protein, its domain structure, post-translational modifications, variants, etc.), a minimal level of redundancy and a high level of integration with other databases</td>
<td>169 MB</td>
<td></td>
<td>2997031</td>
<td>2189259</td>
<td>5</td>
<td>3.55671</td>
</tr>
</tbody>
</table>

**Auction Data**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Uncompressed Size</th>
<th>DTD</th>
<th>Elements</th>
<th>Attributes</th>
<th>Max Depth</th>
<th>Avg Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protein DB</td>
<td></td>
<td>XML (603 MB)</td>
<td>xsd</td>
<td>91</td>
<td>0</td>
<td>5</td>
<td>3.76027</td>
</tr>
</tbody>
</table>

**Figure 3.5: Dataset information**

Disadvantages of this system are the inconvenient search. It is difficult to understand for which tasks one can use this dataset, also lack of information.

Advantages: universal data format XML, for easy conversion to any format for software use.
3.4 Frequent Itemset Mining Dataset Repository

Frequent Itemset Mining Dataset Repository. Often search datasets is a fundamental problem in many tasks of Data Mining, and various approaches to the problem appeared in numerous articles at Data Mining conferences. Although this problem was presented in the context of the market scale, the problem is much wider. Generally speaking, the problem involves the identification of goods, products, symptoms, characteristics, etc., which often occur together in a set of data. As one of the major operations in Data Mining, Algorithms for the FIM can be used as building blocks for other more complex data processes.

FIMI repository open repository of software implementations of Data Mining algorithms and data sets for them, which were accepted at a scientific seminar FIMI. FIMI datasets presented in Figure 3.6.

Frequent Itemset Mining Dataset Repository

The following two datasets were generated using the generator from the EIM Almaden Guest research group. This generator can be downloaded from their website.

- T104C100K (gz)
- E101M100K (gz)

The following datasets were prepared by Roberto Bayards from the UCI datasets and PUMSB.

- chess (gz)
- connect (ap)
- mushroom (gz)
- pairs (gz)
- punish (gz)
- punish, star (gz)

The next dataset was provided to us by Ferenc Boston and contains (anonymized) click-stream data of a funtionary on-line news portal.

- isosatali (gz)

There are three datasets available which were used for the KDD CUP 2000.

They’re described in the paper “Real world performance of association rule algorithms” by Zheng, Kohnke and Mason.

Before you can download the datasets, you are required to click through on an agreement after which you receive a password that will allow you to download the datasets.

- RMS-WebView1
- RMS-WebView2
- RMS-FOS

Figure 3.6: FIMI datasets

The advantage of this repository is that all sets are checked by skilled and credible commission, no more comfort for users was implemented. But this is a logical explanation: The purpose of the repository - storage of algorithmic implementations. The FIM algorithms are sometimes contrary to the requirements. There is a need to fully characterize and understand the algorithmic tasks. It would be interesting to understand why and under what circumstances one algorithm will outperform another. Therefore test methods for a variety of settings are necessary. For example, different data sets, which include dense and sparse, real and synthetic, small and large, from hundreds to tens of thousands of items, thousands of millions of transactions, etc. Data set that is sent to the repository must be accompanied by a detailed description of the algorithm and the set.

To download data sets several conditions are necessary:

- input data should only use ASCII format;
- each agreement is kept on a separate line in the list of elements separated by spaces, and ends with new line character;
- every element is an integral integer;
- The sampling of points should be numbered consecutively, starting with 0, and each operation will be sorted in ascending order.

For a fair comparison, the use of multithreading, advanced pipes, the low level of memory optimization or direct use of hardware, etc. is banned.
3.5 Analytical research
We investigate most popular repositories and found the advantages and drawbacks each of them. UCI repository does not include any methods, but data only. The choice of data is carried out only on filters and data have fixed format. In this repository are lack: preprocessing of data under methods, search personification under problem area. Complexity (brevity description of dataset usage history under a method and problem area) of used data history reading. Advantages of the repository is the downloading speed, i.e. because this repository contains files of .txt format only the data downloading is faster.

The DEA Dataset Repository just as and UCI repository DEA Dataset Repository does not include methods, there is no data preprocessing under methods in it. Files are stored in .xsl format only. Search is only on one of the criteria set, i.e. it is impossible to combine search in several conditions. For not advanced users it is difficult to search in this repository, there is no review of all data sets of the repository. Advantages of this repository is the search in any of criteria.

There are no any methods in XML Data Repository as well as in the repositories set forth above. Inconvenience of search in that there is no understanding, for what tasks it is possible to use this sampling, there is no data preprocessing under methods, there is an insufficient information content by data in it. There is no additional information on data sets and some data sets very large in size. Advantages of this repository is in that data are stored in multipurpose XML format. It is simple for conversion to any other format and is easily applied to program usage. There is no registration in it, therefore it is simple for user to obtain data via http protocol, which allows to use the repository by agents for necessary data searching.

Frequent Itemset Mining Dataset Repository as well as other repositories I researched, this system does not contain methods and accordingly there is no preprocessing under methods. In this system the extremely inconvenient search, there is no displayed additional information about data sets and separate data sets are not accessible for downloading. Advantage of the Frequent Itemset Mining Dataset Repository is that for each data set there is a description of experimental usage.

Analysis of UCI repository, DEA Dataset Repository, XML Data Repository, Frequent Itemset Mining Dataset Repository has shown, that offered model methods and solutions are absent in these repositories. So we developed general concept of Ontology data and metadata exchange repository presented in 3.6 point. (Johnson, 2006; Nyika, 2009)

3.6 Ontology data and metadata exchange repository
Scientific data set Repositories are created for data storage, retrieval, correspondence and processing of data from different subject areas provide a valuable resource for researchers, teachers and students. The storage can help scientists to support their experiments in the field of data mining.

In our repository there are two users: beginner and expert. Each of them has agent. The agent is used in different parts. 
The beginner can:
- find data set using task description (classification approach)
- try to add data and vice versa, choose data domain using dataset

The repository has Coordinator agent (manager). The user agents address to it and deliver tasks to beginner and expert. The repository keeps Data Mining and Machine learning ontology models. Each ontology model has search agent. It receives information (tasks) from coordinator agent. Also dataset ontology model interacts with own dataset agent. And source ontology model interacts with coordinator agent.
4 Multi-agent Intellectual technologies

In this chapter we present detailed information about Multi-agent Intellectual technologies where we discuss basic concepts of agent approach, intelligent agent’s features how it is organized, agent architecture based on BDI model, multi-agent system overview and its classification, which helps us understand its main idea, agent environment overview, agent-oriented system overview for presentation of program part.

Over the past few decades, information technologies have experienced several phases of development. From the first appearance of large computers in the university laboratories to modern laptops or home computers that have the majority of the population. During this period of time, information technology is changed. The language programming and the very principles of programming and software systems were changed. Increasingly, information technologies are being introduced into the modern sphere of human activity in order to improve the quality and ease of labor rights, and it brings great results. Therefore, new demands for information systems are putting. Today, existing technology cannot fully satisfy demands of labor rights. This is due to the rapidly changing business requirements, competition and other factors. But onrush of technology to implement the goals set before them. This continuous development of modern information systems has led to a new level - a system based on agent technology.

4.1 Agent technology

The agents are a new class of software and hardware-software entities that act on behalf of the user. They find and process information, conduct negotiations in electronic commerce systems and services that automate routine operations and support challenges, solutions, collaborate with other software agents in the event of complex problems, thus removing superfluous information from human pressures (Wooldridge M. et al., 1995).

A large number of research laboratories, universities, various businesses and industrial organizations operating in the area of agent systems and technologies. The most prominent research centers are Carnegie Mallon University, University of Massachusetts at Amherst, Bologna University, a number of universities and colleges in the UK as Stanford University, Manchester Metropolitan University. And large corporations IBM, Microsoft, DEC, Apple, Toshiba, Hewlett Packard, etc.

The main directions of scientific research in this area are:
- The agents theory, which treats the mathematical methods and formalisms abstract representations of structure and properties of agents and methods of constructing the logical conclusions of such formal systems;
- Collective agents behavior Method;
- The agents and MAS architecture;
- Methods, languages and communication agents;
- Agents programming languages;
- MAS methods and tools for automated design;
- Methods and means of agent’s mobility.

The areas of practical use of agent technology is information management and computer networks, traffic management, information retrieval, electronic commerce, learning, digital libraries, and many other applications.
4.2 Basic concepts of agent approach
The term «agent» is derived from the Latin verb «agere», meaning «action», «move», «rule», «manage». The encyclopedic dictionary gives the following definition: «agent - a figure, a person acting on the instructions or authority of another». This definition correctly expresses the essence of intelligent agents that can operate autonomously on behalf of its owner (user or another computer system) and to solve various tasks of information processing. For the success work the agent must have sufficient intellectual ability, should have the opportunity to interact with the owner to get jobs and send the results to be guided in their existence and take the necessary decisions (Meyer et al., 2002).

Two basic characteristics - autonomy and purposefulness allow distinguishing intelligent agents from other software and hardware objects (modules, routines and procedures, etc.). The presence of the appropriateness of conduct requires that the intelligent agent has the property of reactivity. This level of intelligence corresponds to reflex behavior of animals. If the intelligent agent has knowledge about the environment, own objectives and ways of achieving them, then the agent can be called intelligent (cognitive).

4.3 Intelligent agent's features
By now, a fairly extensive list of properties that should hold intelligent agents was organized:

- Autonomy, autonomous functioning is the ability to the goals self-formation and functioning of self-their actions and internal state;
- Social ability (social behavior) is the ability to align their behavior with the other agents in a certain environment and rules of conduct through the exchange of messages in the language of communication;
- Reactivity is ability to perceive the state of the environment (environmental performance and a host of other agents) and to respond to changes;
- Pro-activity is the ability to be proactive. It means that agent self-generates goals and acts rationally to achieve them, not just passively responds to external events;
- Basic knowledge is a permanent part of the knowledge of the agent itself, the environment, as well as ongoing knowledge of other agents, which do not change in the life cycle of the agent;
- A Belief is the variable part of the agent knowledge about the environment and other agents that may change over time, but the agent may not know about it and continue to use them to their goals;
- Desires are the attainment stages and / or situation, which is desirable and important for the agent, but may be controversial and will not be achieved at all;
- Goals are the set of states, which are aimed at achieving the current behavior of the agent;
- Intentions are the agent’s obligations to do through their commitments to other agents, or its desire (that is consistent subset of desires, the favorites for one reason or another, and is compatible with its obligations);
- Commitments are tasks that take the agent to request and / or instructions from other agents.

4.4 Belief-Desire- Intention (BDI) agent architecture
Basic knowledge is a necessary component for all the traditional intelligent systems, the conviction must be interpreted in some way in the structure of multi-agent system. Intelligence system (Agent) can be interpreted as the present rules of forming conclusions, the basic scales and weights of criteria, functions, or the benefits and so on. Persuasion has three classes. The first class is the internal belief agent. These algorithms, scripts, evaluation, laid it in the design or made to the operation of the owner or user. The second class includes
inductive beliefs, which arise from the analysis of the environment, emerging production rules of this kind: if there is X, then the conviction Z. The third class is communication of beliefs, attitudes, which appear with other agents, to build production rules, the following: if A says about the X and A is a credible source, then the conviction Z.

4.5 Multi-agent system (MAS)
Multi-agent system (MAS) is a system formed by multiple interacting intelligent agents. The Multi-agent system can be used to solve such problems, which are difficult or impossible to solve with a single agent or monolithic system. The most important characteristics of MAS are situational, autonomy and social flexibility. Situational intelligent agent is defined as the ability to perceive its environment (surroundings) and to act in that environment, if necessary, modifying it for their own purposes (Chopra, et al., 2009; Chopra, et al., 2009; Dastani, et al., 2005). An example of such intelligent agents can be mobile work involved in ROBOCUP competitions, which must interact with the ball, team partners and opponents. The deployment and intentions of other players are not known in advance. Autonomous intelligent agent is ability to interact with the environment without the direct involvement of other agents for which it should be able to control their internal state and actions performed. Flexible agent must demonstrate the quality of sensitivity or foresight (depending on the situation). Responsive agent receives stimuli from their environment and responds to them accordingly. Providently agent does not simply react to the situation in the environment, but also adapts, targeted actions, and chooses the alternatives in various situations. The agent has the property of sociality, if it can properly interact with other software or human agents. Intelligent agent is the only part of the complex process of solving problems in an appropriate environment. Public behavior of agents can take different forms, which can be classified by interaction level (Russell, 2006). Zero level is connectedness, which is outside the owner or user and not accepted by the agents. The first level is coordination. Agents are able to create a situation, which allows other agents to be in the right place at the right time, that as a result of their activities are carried out effectively. The second level is cooperation. Agents admit that their behavior is determined in part by the behavior of other agents when they are jointly trying to achieve a common goal. Such a process for its implementation should be understood by all agents involved in it. The third level is cooperation. It is a real co-operation of agents in the process of implementation, which can benefit everyone. The fourth level is Education Union. It is team-work for a long time during which agents create and maintain conditions of the Union (Weiss et al., 1997).

4.6 MAS classification
The MAS categorization has variety of characters:
- on the location of agents are mobile and fixed;
- on the homogeneity of agents (homogeneous and heterogeneous);
- by way of implementation (software, hardware-software and hardware);
- on the way to solve the problem (closed and open);
- in times of life agents (static and dynamic);
- the way the organization (hierarchical, network and self-organized);
- the nature of the distribution of tasks (functionally-distributed, spatially distributed, functionally and spatial distribution);
- on the principle of evolution (evolving and deterministic).
4.7 Agent environment
One of the most important stages of designing agents is to solve the problem environment. Work environment is essentially "problem for which the intelligent agent is solution". "The notion of critical environment includes such parts as efficiency, environment, actuators, sensors (Performance, Environment, Actuators, Sensors - PEAS). This definition problematic environment there is many options for the environment. But it is still possible to identify a relatively small number of options for the problem environment. They largely determine the most appropriate agent draft and implement the applicable agent.

4.8 Agent-oriented system
Agent technologies and systems development based on multi-approach is a new concept - agent-oriented system (ARS). ARS system should include the following main components: a limited formal language with the appropriate syntax and semantics for describing the internal state of an agent, a programming language for the specification of agents; agentificator (it is turning the neutral components in the programmable agents). In order to understand the principles of the FAC it is convenient to draw a parallel with object-oriented programming (OOP). The object has a name, its own data and procedures. It may consist of several specific sites and, in turn, be part of a larger object. Objects contain fields that can contain data. The field may be simply an attribute or a complex (object). All actions are performed by the PLO through communication. In general, the notion of an object is defined by four key characteristics: encapsulation, abstraction, polymorphism, inheritance. Agent-oriented approach and develops an object becomes a new level of abstraction (Shen W., 1997; Mitkas, et al., 2006). We can say that to some extent, the agent is the object. But the object knows nothing about the nature of relationships between objects and the nature of messages that it did not know and the nature of the world, which it surrounds. The agent is a more complex, active and autonomous unit. In the PLO computing process is understood as the system is collected from modules that interact with one another and have their own ways of handling messages that are received. In turn, FAC clarifies the framework fixing activity modules - agents and changes in their states through the analysis of belief, intentions and responsibilities. The presence of the agent goals design mechanism provides a new level of autonomy. Intelligent agent is not necessarily available to any other agent or user, but simply depends on environmental conditions, including the goals and intentions of other agents. In contrast to the object, the agent can take on certain obligations, or, rather, refuse to execute certain work, saying that lack of competence, employment, and the other task, etc. At the same time, the agent can perform actions such as the creation, suppression and substitution of other agents, to activate functions (both their own and other agents), the intensification scenario of storing the current state of other agents, etc. All this clearly indicates that the agent, as the «active object» or «artificial figure» forms its own conduct. It is at the highest level of complexity in relation to the traditional objects of the PLO.

Agent-oriented architectures and models. The creation of intelligent agents is a difficult task that requires a theoretical foundation for the conceptual representations of agents. This foundation serves a model of intelligent agents, which in many ways describes the knowledge, ways of reasoning, planning, conduct and direct actions of agents. These models can have two ways: first, from the standpoint of analyzing the properties and behavior of agents in the operation of MAS, second, from the perspective of the study and design of agent properties, which determines its internal processes (acquisition of knowledge, development of goals, decision-making, etc.). There are three basic classes of architecture of agent systems and their corresponding models of intelligent agents: deliberative architecture and model; reactive architectures and model; hybrid architectures and models. (Zhang, et al., 2009; Munindar, et al., 2009).
**Deliberative architecture.** Deliberative architecture to determine as the architecture of the agents. It contains the exact symbolic model of the world and the decisions, which are taken on the basis of a logical conclusion. The physical symbol system hypothesis is Theoretical grounds for constructing such models. The hypothesis was formulated by Newell and Simon. The physical symbol system must be physically set realizable of physical entities or characters that are combined in a structure and the ability to run processes that can operate on these symbols in accordance with the symbolically coded sets of instructions. This hypothesis postulates the assertion that system is capable of intelligent behavior in a general sense of the term. According to the interpretation of MR Genesereth deliberate agent architecture should have the following properties:

- contain explicitly provided the knowledge base filled with formula in some logical language, which represents its beliefs;
- operate in the cycle: the perception of the situation the logical conclusion - Actions;
- To make decisions about actions based on a logical conclusion.
- Deliberative agent is something that clearly presents a symbolic model of the world in which decisions (for example, about what actions to perform) are made via logical are conclude based on a comparison with the image or symbolic manipulation. The most common deliberative approaches, a cognitive component contains essentially two parts: a scheduler and the model of the world (Figure 4.1).

![Figure 4.1: Deliberative architecture base](image)

The world model is an internal description of the external environment for the agent and sometimes also includes a description of the agent. Scheduler uses this description to create a plan to achieve the goal of an agent. He is asking atomic actions (operators) that the agent is able to perform their assumptions and their results in the world and initial and target situation. He is looking for a sequence in the space of operators that have not found yet. He transforms the initial state to target state.

End Plan is a list of actions, which passed the artist plans and will perform these actions. End Plan lead to different procedures of low effectors.

In recent decades several realizations of deliberative architecture have been proposed. Most of them were used only in limited artificial environments and only a few have been applied to solve real problems and a very small number brought to the stage of actual corporate programs. One such architecture is the Multi-Agent Reasoning System (dMARS) based on an older system of Procedural Reasoning System (PRS) and uses the conceptual framework of BDI - model output. Models agents and PRS dMARS are examples of the most popular at present paradigm known as the BDI-approach. BDI-architecture typically contains 4 key data structures: beliefs, goals, intentions and plans of the library.

Agent Persuasions correspond to the information agent for the world and may be incomplete and incorrect. Typically, agents in the BDI-model remain in a symbolic form. The desires of agents (or targets) intuitively correspond to the tasks assigned to the agent. For existing BDI - agents need to desire to be logically consistent although the human desires often do not meet this requirement. Agents may not achieve all their desires even if those wishes are contrary to one another. Agents need to fix a set of achievable dreams and to allocate resources to achieve them. Agent’s selected wishes are the intentions. Agent will seek to achieve those intentions until its beliefs and desire or the will with these beliefs do not become more attainable.
For example, in the dMARS each agent has a library of plans that determines the range of possible actions that can be made by an agent to achieve his intentions. Thus, Plans realize the procedural knowledge of the agent. Each plan contains several components. Trigger condition determines the circumstances under which the plan should be viewed as an opportunity for the application. The plan has a context or background. It defines the circumstances under which the plan can begin. Also the plan has the main condition that must be correct in the performance plan. Also the plan has a body. The body may include goals and primitive actions. Events that are perceived by the agent are placed in a queue of events. The inner agent interpreter continuously executes the following cycle:

- survey multi-world and the internal state of the agent and changes the turn of events;
- generate new possible desires (tasks) and find plans whose triggers events included;
- chooses from the set plans included just one plan for implementation;
- puts the desire value to an existing or a new stack, according to sub goal;
- selects the intentions of the stack, reads the plan, which is in the top of the stack and performs the next step of this and executes plan if a step is action if the step is a goal, so it is placed in a queue of events;
- Revert.

Deliberative models developments are attempts by the formalization of new motivational properties and relationships in combination with the behavior and actions of agents (Xaken, 2005). This approach leads to the development of abstract logical models. It is pretend to strict formal description of all relevant properties of rational agents in the specification and verification of MAS.

Construction of such architecture requires problem solving such as the construction of an adequate symbolic description of the real world (it takes into account the complexity of processes occurring over time and existing sites), the logical conclusion organization from the available knowledge, which should lead to specific actions of the agent.

The deliberative models and architectures advantage is the possibility of stricter application of formal methods and traditional technologies of artificial intelligence. Artificial intelligence technologies allow relatively easy to represent knowledge in symbolic. The creation of complete and accurate model of some substantive areas of the real world, a formalization of the mental properties of agents and processes of reasoning in these cognitive structures represent significant challenges for the technical implementation.

**Reactive architecture.** Reactive architectures are finding problem solving. It encounters in the use classical methods of artificial intelligence of agent systems. The founder of this direction is R. Brooks, who both made key ideas behaviorist look at the intelligence:

- intelligent behavior can be established without the explicit character of knowledge;
- intelligent behavior can be created without the express abstract logical conclusion;
- Intelligence is an emergent property of sudden complex systems.

In the real world, intelligence is not an expert system or logical conclusion machine. But intelligent behavior arises as a result of the interaction of an agent with the environment. Instead of modeling the world and reactive planning agents should have a collection of simple behavioral patterns that are responsive to changes in the environment in the form of «stimulus - reaction» (Figure 4.2)

![Figure 4.2: Reactive architecture base](image-url)
The most controversial of the Brook’s principles is the principle of representation. He argues that a clear presentation of the world is not necessary for the implementation of effective agents. Instead, the agent must use the «world as its own model - the continuous treatment to its own sensors is better than an internal model of the world» (Brooks, 1991). Yet agents, in several experiments were proved the ability to handle a limited number of simple tasks in the fields of the real world. But they face problems in carrying out tasks about knowledge of the world that transcend the logical conclusion or from memory. The reactive agents are often made «tough» and have no ability to learn.

The Architecture models. The best known model is the M-agent architecture of MAS. This multi-world (MA-world) includes the agents, agent space and environment, the relationship between agents and environment and relationships among agents. Common definitions of M-agent architecture are as follows: “a” - the agent (“a” is particular agent in MA-world), “and” - many agents that exist in the AI-world, which is called the configuration of agents, “N” - the set of all configurations of agents. Introdued concepts related to the agent type: G - the set of possible types of agents in the MA-world; “a” (ig) is the agent type of g; “A” (g) - a lot of agents of type “g” of the MA-world called as the configuration of agents of type “g”; “N” (g)-configurations of many different types of agents “g”. Space Live of agents is determined by the notion of a resource “r”. The set of resources “R” (Resource configuration), “Rs” – sets of variety configurations of resources in the MA-world. The topology of the live space of “T” determines a lot of places “t” where agents can live and work. Then the structure of space is defined as a pair. Model agent is defined as a structure. The model of the reactive agent architecture includes such things as “t” - agent environment model, “M” - the environment model set (i.e. the agent's knowledge about the environment), “q” - agent purpose, “s”-agent strategy, “S” - the set of strategies of agents called as configuration strategies. Behavior of M-agent architecture is described by the following actions:

- creation of an agent and its placement in the v;
- the agent considers the surrounding space and builds a model of its environment – “t”;
- choose the best strategy “S”, which can be performed;
- if “S” is found, then transition to the next step, otherwise return to monitor the environment;
- implementation of the chosen strategy “S”;
- the agent considers the environment and builds a model of its new environment “t *”;
- adaptation of the environmental conditions and the transition to the best strategy;
- transition to step surveillance environment.

In the M-agent model architecture to provide orientation of the agents with the resources. The relationship of agents and the space is not precisely defined. In this model is not hierarchy of agents is not determined by the logical relationship of agents, no possibility of a logical conclusion on MA-world and the relationship among agents. The behavior of the agent in the M-agent architecture is essentially a struggle for some resources, as well as a goal of the agent is formulated as a function of the achievement of the resource. It does not take into the background account of development. A constructive mechanism does not propose for implementing the strategies. One consequence of these problems is the inability to link high-level specification to implementation of such models in the MAS.

**Hybrid architecture.** Reactive approach allows efficient use of the set of simple scenarios of behavior of agents in the set of reactions to certain events of the environment. But its apparent limitations in the practical impossibility of the whole situation analysis of all possible active agent. Therefore, in most projects and existing systems the hybrid architecture (Figure 4.3) is used. Recently, some researchers recognize that the intelligent agent must have a high level conclusion and the ability of low-level jet. The reactive capacity for current tasks
and the possibility to its logical conclusion for the more complex long-term objectives were used. There are two categories of hybrid agent architectures: Homogeneous architecture uses a common representation and control scheme for the reactions and reasoning, while the multilayer architectures use different representations and algorithms (implemented in the individual layers) to perform these functions.

![Figure 4.3: Hybrid multi-layer agent base architecture](image)

Reactive component reflects perceptual incentives for primitive actions. Deliberative component is a symbolic conclusion to control the behavior of reactive components, such as the situation changes a lot of rules (actions). In some architecture deliberative component is directly linked to sensors and effectors agent. When the layered hybrid agent architecture is designed we must obtain answers to critical questions: Is only one jet and one deliberative level enough? Do we have to introduce more levels? How a cognitive workload is split between the levels? How should the components interact at different levels? When the agent should act and when talk (i.e. how to divide the algorithm of decomposition)?

Among the most popular in our time of hybrid architectures can be noted such as a cyclic machine architecture proposed by Ferguson; Glair (Grounded Layered Architecture with Integrated Reasoning) architecture, or multi-architecture with generalized conclusion; DYNA architecture, architecture InteRRaP. In order to more fully acquaint with hybrid architectures consider proposals Ferguson, cyclical machine architecture.

Cyclic machine Architecture includes three levels: level jet (set of rules «situation-action»), planning level (the main component is hierarchical), part-time planner and modeling levels. The aim of reactive level is to ensure rapid response to events in real time. The main objective of the planned level is to generate and execute plans to achieve long-term goals of the agent. Finally, the purpose of the simulator is to identify and anticipate situations of potential conflict between the objectives of the agents. Then propose actions to exit from these conflicts (Figure 4.4).

Each level is independently associated with the sensors and effectors and act as head to the agent. As a result of this action will often conflict with each other. Conflicts are solved with the help of suppression rules. On the side of the receptors has a policy of censorship that filters sensory data so that each receives the appropriate level of sensory data. These rules may be organized in different ways. This will depend on the architecture and capabilities of the agent, which is being developed. The important point is the consistency of the censorship rules. Responsibility is rests to the developer's architecture and intelligent agent.
Figure 4.4: Agent structure of cyclic-machine architecture

Messages are divided into two types: passive transfer of information (reactive level reported to the model the fact that the world should pay attention) and actively changing the control solutions at other levels (model level raises the design level to generate a plan for new task. Levels operate in parallel, but the synchronous (using internal clocks agent) (Jennings N. et al., 1995), (Shen W. et al., 1996).
5 Ontology models development

In this chapter we present ontology models research and development. We created several ontology models, which contain description about their structure and usage in ontology data and metadata exchange repository.

5.1 Implementation
Search algorithm is based on ontological models and ranking results. Implementation of access to intelligent agents based on web-services, implemented on the basis of programming paradigms such as dependency injection and control inversion. There were a variety of access through the implementation of web-projects and implemented a more flexible, robust architecture. During the work we analyzed agent-oriented approach methods and put the template design.

During the work we offered not only general model of the search module, but its detailed architecture and implementation. Architecture of intelligent search agent was proposed and the possibility of its use was investigated in detail. The design patterns in agent-oriented approach were developed and implemented.

The above mentioned drawbacks have a point to create a new public information repository to store datasets using intelligent agent and ontological approach for storing, conversion, search, add, description, selection of the required information for researchers’ needs in the field of Data mining and Machine Learning. Using the Protégé 3.4 we created an ontology model of Data mining methods, an ontology model of the user, a model of the resource (W3C, 2004).

A base standard was chosen the standard SDMX Standards Version 2.0 and the main parameters of the Statistical European Repositories were taken. The interaction between the ontological models is based on intelligent agents: coordinator agent, resource agent, search agent, a user agent. The agent approach has been implemented by multitechnology JADEX. We use intelligent software agents. This is a new class of software systems, which acts either on behalf of the user, or on behalf of the system. They are, in fact, a new level of abstraction, different from the usual abstract type - classes, methods and functions. For practical implementation of these agents JADE offers to programmer-designer of agent systems the following possibilities: FIPA-compliant Agent Platform, which includes system agents AMS, ACC and DF; Multiple Domains support – DF agents and so on (IEEE Computer Society standards organization, 2006; Bellifemine, et al., 2006).

5.2 Ontology representation
In recent years the development of ontologies are formal descriptions of explicit terms for business and relations between them. In the World Wide Web became commonplace ontology. Ontology in the network range from large taxonomy that categorize websites (Yahoo! website) to products and their characteristics (like on the website Amazon.com). Consortium WWW (W3C) develops RDF (Resource Description Framework). The RDF is language of encoding knowledge on Web pages. It makes knowledge understandable to electronic agents to search information. Now many disciplines develop standard ontology that can be used by experts in subject areas to share and annotate information in their field. For example, in medicine large standard structured dictionaries such as semantic web unified medical language system (the Unified Medical Language System) were created. Also large ontology appears general intent. For example, the UN Program for Development (the United Nations Development Program) and the company Dun & Bradstreet combined efforts to develop ontology UNSPSC. It provides terms for goods and services. Ontology defines a common vocabulary for researchers, who need to share information in the subject area. It
includes machine-interpreted formulating the basic notions of domain and relations between them. Ontologies are developed for joint use by people or software agents common understanding of data structures for possible reuse of knowledge in the subject area, to make assumptions explicit in the subject area, to separate knowledge in subject area of operational knowledge, the analysis of knowledge in subject area.

5.3 Program - instrumental method of implementation of the ontological model
The tool Protégé 3.4 for “data and metadata exchange repository” was selected. It was developed at Stanford University (USA) (Gennari J.H. et al., 2002). Protégé 3.4 is meta-tool. It helps users to create a system of acquisition knowledge for a particular subject area and experts can use these systems to enter and view the information contained in electronic databases of knowledge. The modular architecture of Protégé 3.4 very expands class of systems that can be collected for certain tasks on the acquisition of knowledge and making the future of knowledge acquisition can be better adjusted in accordance with certain requirements of end users. The Protégé 3.4 developers say: “The system is open software. It is difficult to calculate the number of users...” Now the list on the Protégé, nearly 9,000 subscribers, and website Protégé registered over 20,000 users (we can download the Protégé without registration). You can download 85 different plugins for Protege from the site. Protégé user community is very active and has representatives in more than 100 countries. The functional editor is inextricably linked to the specific for the ontology model and knowledge arising from the classification scheme vocabulary. The editor has a graphical interface that provides a visual edit mode. Graphical interface is implemented on the basis of standard software Object TreeView, a significant addition of additional functionality - mainly in the search, input and control logic. Ontology editor Functionality is:

- View and search: supports viewing grid, standard types of search time;
- editing (input, correction, deletion);
- logical control in the introduction: the introduction of technology almost completely eliminates the violation defined description schemes;
- functionality testing: writing queries;
- interaction with other ontology (import - export, mainly using communicative presentation formats).

5.4 Ontology source model (dataset ontology model)
Information about “data and metadata exchange repository” is stored in the ontological models form. One of the main classes of this model is «data set» (DataSet). Each separate instance of this class contains information about the data set to this information include name, analysis method, short description, information about its creators and more. This class contains several classes that belong to its structure: DataSetFile and Judge. Class DataSetFile contains information about the sample that covers this data set, but Judge class contains information about the evaluation of the different set of moderators.

5.4.1 Ontological source models development
The Ontologies are developed and can be used in solving various problems, including joint use of people or software agents to possible accumulation and reuse of knowledge in the subject area, to create models and programs that operate ontology, but not rigidly defined data structures, analysis of knowledge in the subject area. For a more intelligent synthesis of information systems section must define ontology, which should describe the terminology used in the contents of set rules for the use of these terms in the context of other terms.

The basic building block of dataset model is an assertion that represents: resource named property and value. In RDF terminology these three statements are respectively: subject,
predicate and object (W3C, 1999). Show description of the dataset source in the environment of ontologies Protégé 3.4. Classes and attributes of selected classes are created and presented in Figure 5.1. In the development of ontological models of the resource repository was allocated 3 classes. Here more detail the selection process classes. First of all area and scope of ontology were defined. Then important terms of source ontological model of "data and metadata exchange repository": sample, method of analysis, attribute, subject area, data set description, the dataset file, name, type, articles that refer to the dataset, keywords, author, date of creation. We highlighted three classes and a set of slots in the ontological model of the resource:

- DataSet;
- DataSetFile;
- Judge.

Figure 5.1: Ontology source model Classes and attributes in the protégé-3.4

In this thesis ontology source model are described by three classes.
### 5.4.2 The ontological models source of classes description

We present slots description of DataSet class in Table 5.1. It is serving for dataset description.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Power</th>
<th>Presence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>Introduction (short description)</td>
</tr>
<tr>
<td>AnalysisMethod</td>
<td>String</td>
<td>Multiple</td>
<td>Mandatory</td>
<td>Analysis Method (refer to ontology elements of dataset analysis methods)</td>
</tr>
<tr>
<td>Area</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>Data domain</td>
</tr>
<tr>
<td>AttributeAmount</td>
<td>Integer</td>
<td>Single</td>
<td>Mandatory</td>
<td>Attribute number</td>
</tr>
<tr>
<td>AttributeInfo</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>Attribute information</td>
</tr>
<tr>
<td>AttributeType</td>
<td>String</td>
<td>Multiple</td>
<td>Mandatory</td>
<td>Attributes type</td>
</tr>
<tr>
<td>CitedPaper</td>
<td>String</td>
<td>Multiple</td>
<td>Optional</td>
<td>Articles refer to datasets.</td>
</tr>
<tr>
<td>Creators</td>
<td>String</td>
<td>Multiple</td>
<td>Mandatory</td>
<td>List of dataset creators: refers to user ontology element</td>
</tr>
<tr>
<td>DataSetInfo</td>
<td>String</td>
<td>Single</td>
<td>Optional</td>
<td>Dataset description</td>
</tr>
<tr>
<td>DataType</td>
<td>String</td>
<td>Multiple</td>
<td>Mandatory</td>
<td>Data type</td>
</tr>
<tr>
<td>DateDonated</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>date of data loading or last date of update</td>
</tr>
<tr>
<td>DownloadAmount</td>
<td>Integer</td>
<td>Single</td>
<td>Mandatory</td>
<td>downloading</td>
</tr>
<tr>
<td>DSFiles</td>
<td>Instance</td>
<td>Multiple</td>
<td>Optional</td>
<td>Datasets files</td>
</tr>
<tr>
<td>InstanceAmount</td>
<td>Integer</td>
<td>Single</td>
<td>Mandatory</td>
<td>Number elements in the dataset</td>
</tr>
<tr>
<td>KeyWord</td>
<td>String</td>
<td>Multiple</td>
<td>Optional</td>
<td>Keywords</td>
</tr>
<tr>
<td>RelevantPapers</td>
<td>String</td>
<td>Multiple</td>
<td>Optional</td>
<td>Relevant papers</td>
</tr>
<tr>
<td>SolutionMethods</td>
<td>String</td>
<td>Single</td>
<td>Optional</td>
<td>Method of solution</td>
</tr>
<tr>
<td>Status</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>Dataset status (new, low, middle, high)</td>
</tr>
<tr>
<td>DatasetMark</td>
<td>Float</td>
<td>Single</td>
<td>Optional</td>
<td>Average dataset estimation</td>
</tr>
<tr>
<td>Title</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>Title</td>
</tr>
</tbody>
</table>

*Table 5.1: DataSet class slots*

In Table 5.2 we present slots description DataSetFile class for files datasets description.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Power</th>
<th>Presence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileDescription</td>
<td>String</td>
<td>Single</td>
<td>Optional</td>
<td>File description</td>
</tr>
<tr>
<td>LastModified</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>The last load or modification date</td>
</tr>
<tr>
<td>Size</td>
<td>Float</td>
<td>Single</td>
<td>Mandatory</td>
<td>File size, kb</td>
</tr>
<tr>
<td>Title</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>File name</td>
</tr>
<tr>
<td>Extension</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>File type</td>
</tr>
</tbody>
</table>

*Table 5.2: DataSetFile class slots*
In Table 5.3 we present slots description Judge class for description of estimation dataset.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Power</th>
<th>Presence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login</td>
<td>String</td>
<td>Single</td>
<td>Mandatory</td>
<td>Person, who tick off (invocation ontology element)</td>
</tr>
<tr>
<td>Comments</td>
<td>String</td>
<td>Multiple</td>
<td>Mandatory</td>
<td>User comments about datasets</td>
</tr>
<tr>
<td>Mark</td>
<td>Integer</td>
<td>Single</td>
<td>Mandatory</td>
<td>Estimation</td>
</tr>
</tbody>
</table>

Table 5.3: Judge class slots

After ontology model source determination Protégé system allows to convert Protégé project to RDF model.

5.5 Ontology data mining model
Ontology data mining model is the exact specification of the subject area. It provides a vocabulary for presenting and sharing knowledge about methods of analysis and methods of deduction and many relationships established between terms in the dictionary. One of the advantages of using this ontology is a systematic approach to the study of the subject area. It is achieved: systematic (ontology presents a holistic view of the subject area); monotony is material (represented in a single form is much better perceived and reproduced), scientific development (construction of ontology can restore missing logical links in their entirety).

5.5.1 Ontological data mining models development
There are two levels on which ontologies are used to support data processing: domain ontologies and task ontologies. Domain ontologies are used to describe knowledge from the domains relevant to the particular task (Figure 5.2). The first step in the ontology development is the definition of the domain and scope of the ontology itself: in our scenario the ontology will cover the Data Mining domain. To build a consistent ontology model it is necessary to establish for what we are going to use the ontology and for what types of questions the information in the ontology should provide answer. The choice of how to structure ontology determines what a system can know and reason about. We have built our ontology through a characterization of data mining methods that is classified on the basis of some parameters useful to select the more ones method to solve a KDD problem. Repository determines characteristics of the data and of the desired mining result, and enumerates the DM processes that are valid for producing the desired result from the given data. Then the Repository assists the user in choosing processes to execute, for example, by ranking the process (heuristically) according to what is important to the user. Results will need to be ranked differently for different users. A different user may want to minimize run time, in order to get results quickly. There are other ranking criteria: accuracy, cost sensitivity, comprehensibility, etc., and many combinations thereof.
Figure 5.2: Data mining ontology, where \( \rightarrow \) property relation of concepts, Subclass \( \rightarrow \rightarrow \) relation of concepts

To solve problems related to data analysis in the presence of random and unpredictable effects, mathematicians and other researchers over the last two hundred years produced a powerful and flexible arsenal of methods, collectively called mathematical statistics. During this time extensive experience was gained in the successful application of these methods in different spheres of human activity from economics to space research. And under certain conditions these methods allow for the optimal solution. For example, one of the problems solved in the radiolocation is the known signal detection in background additive interference in the form of white noise. Mathematical statistics methods solve this problem successfully. It is difficult to imagine the need for other approaches to solving this problem.

Because knowledge is personal in nature, the same subject area can be described by different ontologies. This is particularly true of domains that are not formalized or when there are many contentious issues. In this work one of the problems is the task of ontology development methods is data mining. Certainly a good practice is to use already existing ontologies and a good specialist should be able to quickly find existing and already proven any ontology or an algorithm, rather than spend time on developing new. The fact is that ontologies are not clearly structured and formalized. Now a lot of online ontologies and of course they are all correct. But research of existing data mining ontologies did not give a satisfactory result.
Therefore a new ontology was developed. Analysis of knowledge in the subject field of Data Mining is quite possible because there is a declarative specification of terms. Formal analysis of the terms will be extremely valuable as when to reuse the developed ontology so in its expansion. The reason for the development of ontology data analysis provides an Analysis Method slot in DataSet class of developed ontological model resource. It contains data mining methods that are under all this set of statistics. Ontology with a set of individual instances of classes forms a knowledge base. In fact, in this case it is difficult to determine where the ontology ends and where the start of knowledge base. Ontological model was presented in Figure 5.3.

5.6 Ontology user model
Ontological approach is offered for creation of model of user for intellectual repository “data and metadata exchange repository”. This approach allows taking into account the collection of concepts and connections between them, having a place at interaction of the user with our repository. Ontology user model is the model for data structuring. It stores information about user. User model is obviously for our repository with different levels of training for work with a computer, with a variety of mental, psychological and physiological capabilities (Cargar, 2008; Waltz, 2008).
5.6.1 RDF model
The namespace http://dmr.kture.ua/dataset/ conversion is defined. The part of RDF model is shown in Figure 5.4. Also ontology user model and data mining ontology model was developed. Ontology methods are a classification of data mining. The user ontology has two abstract classes: Account and Person. Class Account represents user as a logical entity of user’s system. Class Person represents person as a person that uses this system.

The real value of RDF cannot be evaluated until it is used for internal purposes of a given program. The benefits of implementing RDF will be when it becomes a means systems interaction, data exchange, when the machine will get the ability to combine information obtained from different sources, thus getting some new information. The more applications on the Internet can work with data the higher will be their value.

The obtained RDF-model is the metadata of experimental datasets. It will further develop multi-system based on metadata datasets work.

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE rdf PUBLIC 
  'http://www.w3.org/1999/02/22-rdf-syntax-ns#' 
  'http://dmr.kture.ua/dataset/'>
<!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema'>

<rdf:RDF xmlns:rdf='&rdfs;' 
  xmlns:dataset='&dataset;' 
  xmlns:rdfs='&rdfs;'>
  <rdf:Property rdf:about='&dataset;Abstract' 
    rdfs:label='Abstract'>
    <rdfs:domain rdf:resource='&dataset;DataSet'/> 
    <rdfs:range rdf:resource='&rdfs;Literal'/> 
  </rdf:Property>
  <rdf:Property rdf:about='&dataset;AnalysisMethod' 
    rdfs:label='AnalysisMethod'>
    <rdfs:domain rdf:resource='&dataset;DataSet'/> 
    <rdfs:range rdf:resource='&rdfs;Literal'/> 
  </rdf:Property>
</rdf:RDF>
```

**Figure 5.4: Part of RDF model**

5.6.2 Ontological user models development
In Figure 5.5 user ontology model in the system Protégé 3.4 is represented. Protégé system has the following possibilities: tabs for ontology replenishment, functional expansion modules, generation of knowledge acquisition module requests and the logical deduction module.
This ontological model includes two abstract classes: Account and Person. Class Account represents the user as the logical nature of the user system. Class Person represents the user as the person using the system. Experienced and Beginner classes are beginner and advanced user respectively. Admin slots class match Experienced slot class.

5.6.3 The ontological user models of classes description.
In Table 5.4 the description of Address slots class is shown.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Power</th>
<th>Presence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>country</td>
<td>String</td>
<td>1</td>
<td>Mandatory</td>
<td>country</td>
</tr>
<tr>
<td>city</td>
<td>String</td>
<td>1</td>
<td>Optional</td>
<td>city</td>
</tr>
</tbody>
</table>

Table 5.4: Address slots class

In Table 5.5 University slots class is present. It works for base University description.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Power</th>
<th>Presence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>1</td>
<td>Mandatory</td>
<td>University name</td>
</tr>
<tr>
<td>address</td>
<td>Address</td>
<td>1</td>
<td>Optional</td>
<td>University address</td>
</tr>
</tbody>
</table>

Table 5.5: University slots class
In Table 5.6 Preference slots class is present. It works for user interest and search requests description.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Power</th>
<th>Presence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interest</td>
<td>DataMiningMethod</td>
<td>*</td>
<td>Optional</td>
<td>Data format</td>
</tr>
<tr>
<td>search</td>
<td>String</td>
<td>*</td>
<td>Optional</td>
<td>A lot of user search requests</td>
</tr>
<tr>
<td>searchHistory</td>
<td>SearchHistory</td>
<td>*</td>
<td>Optional</td>
<td>A lot of user search requests results</td>
</tr>
</tbody>
</table>

**Table 5.6: Preference slots class**

The abstract Account slots class is present in Table 5.7.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Power</th>
<th>Presence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>password</td>
<td>String</td>
<td>1</td>
<td>Mandatory</td>
<td>password</td>
</tr>
<tr>
<td>created</td>
<td>String</td>
<td>1</td>
<td>Mandatory</td>
<td>Date of creation</td>
</tr>
<tr>
<td>email</td>
<td>String</td>
<td>1</td>
<td>Mandatory</td>
<td>e-mail</td>
</tr>
<tr>
<td>preferences</td>
<td>Preference</td>
<td>1</td>
<td>Optional</td>
<td>Information about preferences</td>
</tr>
<tr>
<td>title</td>
<td>String</td>
<td>1</td>
<td>Optional</td>
<td>display name</td>
</tr>
</tbody>
</table>

**Table 5.7: Slots of abstract class Account**

The Account class is user representation base.

The slots of abstract class Person are present in Table 5.8. Its base class is Account. Class Person is base for Beginner and Experienced classes. Beginner class has the same slots as class Person. Slots of Experienced class are present in Table 5.9.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Power</th>
<th>Presence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>first_name</td>
<td>String</td>
<td>1</td>
<td>Optional</td>
<td>Name</td>
</tr>
<tr>
<td>last_name</td>
<td>String</td>
<td>1</td>
<td>Optional</td>
<td>Surname</td>
</tr>
<tr>
<td>gender</td>
<td>Symbol (Male, Female)</td>
<td>1</td>
<td>Optional</td>
<td>Sex (male\female)</td>
</tr>
<tr>
<td>university</td>
<td>University</td>
<td>1</td>
<td>Optional</td>
<td>Information about university</td>
</tr>
</tbody>
</table>

**Table 5.8: Slots of abstract class Person**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Power</th>
<th>Presence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>speciality</td>
<td>String</td>
<td>*</td>
<td>Mandatory</td>
<td>speciality</td>
</tr>
</tbody>
</table>

**Table 5.9: slots of Experienced class**
6 Intelligent search agent design and development

As one of basic concept in this thesis is search agent. This chapter presents the detailed description of search agent development.

The knowledge (beliefs) mechanism is used by agents to store the internal data. There are two types of knowledge: atomic knowledge (belief) and set of knowledge (beliefset). The objective (goals) and plans (plans) mechanism are used by agents to achieve the assigned tasks. All actions of agents have an effect on their objectives. Depending on the current objective, the agent executes either the proper plan or series of plans.

6.1 Agent implementation

Startup each agent of the system executes WebRegistrationPlan, which registers the agent in the system that allows its later calling through a web service. Separate web-service is provided for each agent in the system, including methods, initiated the relevant aims of the agent.

The agents execute plans performing the aims. They interact with each other and return the result to the web service. Each class of agent is formed by agent platform according to the agent description. Agent platform analyzes the agent description and verify the availability of classes in the building plans of the agent. Thus, the agent class diagram includes a set of plans classes of the agent, as well as if the agent uses a capability there must be the reference both to the description file and to the classes.

The objectives and plans of agents correspond to each other using the following rule: if an agent has a goal with the name xxx, the plan will have name xxx_plan. All classes of planes also have the word Plan at the end.

The parameters are sent into the plans via a goal mapping mechanism, i.e. parameters that have been set for the objectives of the agent. They are displayed on the plan parameters.

The following rule was also approved for the objectives, if the objective returns one parameter, the name of this parameter is result. Interaction between agents is a standard Directory Facilitator mechanism. When the agent starts its operating it executes WebRegistrationPlan and objective of an agent registration to receive messages from other agents of the system. When the agent is searching for the agents of the same type it applies a Service Name mechanism.

6.2 Intelligent search agent

For searching in the data and metadata exchange repository we have to develop a search module. It would consider the current state of system and different searching criteria to adopt any strategy of search. One of the most suitable solutions to this problem is the intelligent agents based on goal. This intelligent agent will act not just in reflective way when a request came, but would decide what actions are needed to achieve its goals in terms of the current state of environment. This agent is not able to supervise the environment, where it’s executed, in full.

In the search module of “data and metadata exchange repository” is set problems such as simple and advanced search or personal search. On the other hand, the search agent is used in the multi-agent environment and agent needs to communicate between it-self and other agents as well as to exert the medium, where it’s executed. From these two points of view the functionality of search agent may be divided into functionality in terms of user and functionality in terms of other agents and execution environment.

Functionality in terms of user should include the following basic set: to execute a simple search only for non-authorized user, to execute the various searches for authorized user and to provide useful services for the search (Figure 6.1).
Both authorized and non-authorized users may execute the simple search, in both cases there will be shown the most popular data sets in the repository or the most popular queries (queries most often made by users in the repository) and the results may also be hints as content search queries (queries correlated with the current ones). But still the authorized user has more privileges in comparison with non-authorized one. The following is available for authorized user: advanced search (search by various data set criteria). When agent uses the search there is displayed some of recent queries. Information about user’s requests and their results are stored using a personal agent and will be used further to provide user with more relevant results considering his previous requests.

![Diagram 1: The use case diagram in terms of user](image1)

On the other hand, the search agent must interact with other agents to successfully achieve the goals. To render the useful information to them and to request the necessary information form them or to request them to provide a service. Figure 6.2 shows the use case diagram in terms of interaction between agents.

![Diagram 2: The use case diagram in terms of interaction between agents](image2)
Personal Agent of user obtains the results from the search agent and the request itself, as well as to take the transition sequence of user until the user will find the necessary dataset. This information will be stored by the private agent in order that the search agent could further use it.

6.3 The search agent goals
The search agent is based on the concept of goals. You can say that the main objective of the search agent is information search, but such a goal is very abstract. You need to shape this goal. Thus it may be divided into two simple goals: simple search and advanced one. We also should to divide the goals into subgoals. For example, defining the significant objectives, the person thinks what he has to do to achieve these objectives and subdivided them into more local ones builds the sequence of his actions. In the same way, you can divide the goals into subgoals and build a hierarchy of goals and actions, as well as assign the extra goals, which in complex will lead to the objectives achievement – relevant search results. After analysis of the search agent functions there can be identified several goals that will be explicitly or not explicitly included to the primary objective search (Figure 6.3). Goals explicitly included to the main goal are simple and advanced search. Such goals as «Get the popular datasets» and «Get contextual queries» can not be completely referred to the search goals, but generally they are subsidiary objectives that can help user in his search.

The simplest search strategy is shown in the «simple search» goal (Figure 6.4). This search option is available both for authorized and non-authorized user. But some of the subgoals of this strategy may vary depending on the state of the user in the system. Let us consider the option strategy when the user is not authorized. In this case, the goal «Simple Search» would include «Search data model», «Gradation results by popularity», «Saving the query string». The Goal «Search data model» has a subgoal, «Create a request to the data model», which task is to form a request to ontological model.
Advanced search is a more complex strategy, where you must consider some different states of the environment and interact with the personal agent of user (Figure 6.5). At this option the subgoal «Formation query data model» is based on the previous queries and user preferences. In this case, the search agent interacts with the personal agent to get the previous requests. After the search on a model the search agent requests a personal search agent to save data such as query string and the results were found. The agent search goal is ranking results by their popularity in the system. This allows the user to provide the most relevant data results. The ranking may be changed as the result of relevant data search increasing, which includes a search agent.
6.4 Search agent outline

The basic Jadex concept is the goals. But the goal in this system is more abstract concept. The Jadex-agents use plans to achieve the goal. Almost every goal of search agent correspond a plan (Figure 6.6). All plans of the agent are an extension of another one, more general, AbstractCommunicationPlan plan. This plan includes methods and information necessary for communication between agents. This class extends another significant type - AbstractDBAccessPlan plan that includes the functionality for database and ontological models.

The SimpleSearchPlan class includes logic of simple query processing from the user. This plan runs required subgoals of the search agent and performs the necessary steps before to execute the simple search goal and also the actions after implementation of this goal. The ExtendedSearchPlan contains the logic of advanced search processing. This plan contains a much more complex logic than the previous one, but the basic principles are the same. Both of these plans are the heirs of an AbstractSearchPlan base class. It contains the functionality general for these both plans.
6.5 Search agent ADF
The agent can not exist without ADF file. ADF file is the main part in the Jadex. This file describes the search agent. It describes goals, plans and knowledge of the search agent. The Figure 6.7 shows the XML structure that contains a partial realization of the search agent goals in the search.agent.xml file. These goals describe the parameters that must or may be transferred to the search agent to successfully complete the target (objective). The search agent has several kinds of goals. The main are goals, which must be achieved, for example, simple or advanced search. Another type of goals, for example, is aims helping to maintain a certain status. For example, a goal that processes the statistical data on the most popular queries to the system is implemented periodically. Simple and advanced searches are described in agent ADF file as goals like achievement. These objectives are directed to the reaching of alternate abstract state methods. The search agent satisfaction is a condition in which the results was found and processed.

```
<achievegoal name="simple_search" exported="true">
  <parameter name="data" class="Object"/>
  <parameter name="sessionInfo" class="Object" optional="true"/>
  <parameter name="result" class="Object" direction="out"/>
</achievegoal>
```

Figure 6.7: The search agent goals description

6.6 Search agent ”Belief”
The search agent retains some knowledge to successfully fulfill the search agent goals and to respond to environmental change. This knowledge also has an effect on agent motives and may influence the set goals of agent. The Search Agent saves the following information:
- query to the search subsystem;
- requests to the system;
- the most popular queries to the system;
- information about the user session for which transaction the agent was created;
- simple rules of relationships and query result;
- history of searches;
- user preferences.

The internal information for the search agent inside is also the data for database access or other ontological models repository. Table 6.1 describes the knowledge of the search agent.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>searchQuery</td>
<td>Belief, String</td>
<td>Search requests to the search subsystem</td>
</tr>
<tr>
<td>searchQuerys</td>
<td>Belief, List&lt;String&gt;</td>
<td>Query to the system</td>
</tr>
<tr>
<td>popularQuerys</td>
<td>Belief, List&lt;String&gt;</td>
<td>The most popular queries to the system</td>
</tr>
<tr>
<td>sessionInfo</td>
<td>Belief, SessionInfo</td>
<td>Information about the user session for which transaction the agent was created</td>
</tr>
<tr>
<td>rules</td>
<td>Belief, ProductionRule</td>
<td>Simple rules of relationships and query result</td>
</tr>
<tr>
<td>userInterests</td>
<td>Beliefset, String</td>
<td>User preferences</td>
</tr>
</tbody>
</table>

**Table 6.1: The search agent knowledge description**

### 6.7 Search agent interaction with other agents

One of the key advantages of multi-agent technology is the communication between agents. The search agent interacts with other agents in the system to provide the user with useful contextual information, to find orientation character for each user individually. With this purpose the search agent interacts with the profile and manager agents and requesting the information about the user, his preferences, query history, and the possible results of the queries (if such a request has been made, he can use the results of previous search). Table 6.2 lists some of the events that are in the search agent.

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction/Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_search_history</td>
<td>send/request</td>
<td>Queries history of user requests</td>
</tr>
<tr>
<td>get_search_history_results</td>
<td>send/request</td>
<td>Request of search results history</td>
</tr>
<tr>
<td>get_interests</td>
<td>send/request</td>
<td>The request to receive user preferences</td>
</tr>
<tr>
<td>add_to_search_history</td>
<td>send/info</td>
<td>The messages to save history requests</td>
</tr>
<tr>
<td>add_search_history_result</td>
<td>send/info</td>
<td>Saving of search results history</td>
</tr>
</tbody>
</table>

**Table 6.2: The search agent events**

### 6.7.1 Search agent Scenario

Intelligent agents – a new class of software and hardware entities. Such substance acting on behalf of the user to find and process information. Based on current knowledge and events of environment search agent selects a plan to achieve its goal – to find the most relevant data at the user's request.

Such a structure (illustrated in Figure 6.8) enables the agent to effectively search in the repository.
6.7.2 User agent Scenario
The base information unit of the personal agent is ontology model of user (section 5). At the level of agent conception about the user is object model of user ontology. The main objective of the personal agent is to transfer information about users to other agents and to transfer the necessary information to user from the other system agents. So, personal agent should be able to form answers to queries from other agents of “data and metadata exchange repository” system and to modify the user profile during his work with the system. In accordance with the information and ontology model of user the personal agent should be able to form answers to questions related to user. We can allocate the following two partitions of information about user: personal information about user, information about current goals of user. In general case the personal agent should be able to respond the following questions: what is user name, what is his e-mail address, what language user prefers, what are the current goals of user; is user advanced or beginner (naïve, simple), what academic institutions the user belongs to; what are localization preference of the user; are the interests of user coincide with other users interests in the system; what are the recent requests of the user. Here the personal agent applies the developed information and ontology model of the user for questions, which can be requested by other agents while interaction with personal program agent during the work of user with the system.

The User Agent is created after the user authentication. Figure 6.9 shows the User Agent functionality.
When authentication and authorization are completed the user agent retrieves its knowledge information about users that later allows the user and other agents to access this information quickly. The user agent stores this information in its knowledge when active user is in the system and before the work cessation; the agent unloads this information into the database. The User Agents in the system as much as users have passed authentication at the current period. If the user does not operate with agent over 30 minutes, the user agent removes the search agent and itself from the system.

The user sets the following tasks before the user-agent:

- user personal data storing;
- changing of user data;
- preserve a user’s search query to the system;
- conservation of user activity in the system;
- tracking the status of the user in the system;
- retention of the data sets loaded into the system;
- retention of the data sets unloaded from the system;
- User communication with other users of the system.

The user agent inside information. The user agent stores information in its knowledge:

- personal information about the user;
- type of user;
- history of search queries;
- history of user queries results;
- new search results;
- interests of the user.

Table 6.3 shows the user agent knowledge and its type. All knowledge of the agent is loaded at the appropriate request. All data are stored in the database or synchronize with it at the end of agent operation.
The user agent knowledge may be also dynamically expanded during the agent life.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>Belief, Account</td>
<td>personal information about the user</td>
</tr>
<tr>
<td>account_type</td>
<td>Belief, Class</td>
<td>type of user</td>
</tr>
<tr>
<td>Search_history</td>
<td>Belief, History</td>
<td>history of search queries</td>
</tr>
<tr>
<td>Search_results_history</td>
<td>Belief, SearchHistory</td>
<td>history of the user queries results</td>
</tr>
<tr>
<td>new_search_results_history</td>
<td>Belief, SearchHistory</td>
<td>new search results</td>
</tr>
<tr>
<td>Interests</td>
<td>Beliefset, String</td>
<td>interests of the user</td>
</tr>
</tbody>
</table>

Table 6.3: The User agent knowledge

The input data for the user agent. Personal agent receives the following data from the user:
- personal information needed to be updated;
- information about the loaded data sets from the system.

End users of the system interact with the user agent. It invokes the Web service from user interface of user agent, which on the basis of the information about current client session sends an invocation to the personal agent.

Personal agent receives the following information from the search agent:
- user queries;
- list of user queries results.

Search Agent informs the user Agent about the results of queries that user has implemented. It allows during the further use of the system to get quick access to these data without additional requests to the database.

Personal agent receives from the source agent the information about data sets loading by user into the system.

A source agent informs the user agent when the user downloads data sets into the system. In its turn the user agent saves itself knowledge references to these data. It helps the user to have quick access to editing of the system data sets.

The user agent date-line. Personal Agent provides the user with the following information:
- user's personal information;
- a user's search query;
- history of the search queries results;
- a list of data sets downloaded into the system;
- a list of data sets downloaded from the system;
- a list of users, who have similar interests and are or registered in the system.

The user has quick access to data sets, which he unloads from the system obtaining information from the user agent.

The users of the system can know about each other on the basis of the data stored by user agent. For example, advanced users, who have the scientific interests in their personal information will be able to find the results of queries and activity from other users having the similar interests.

The agent sends requests to the agents of other users and based on the obtained results forms the information that he interested in to search users, who use the system.
The personal agent provides the search agent with the following information:

- user interests;
- list of user requests;
- list of user queries results.

The user agent interaction with other agents. Personal agent interacts with manager (coordinator) agent to make the task: the request sending to change the status of the user.

Personal agent interacts with the search agent for the following tasks:

- to obtain information about the search query;
- to obtain information about the query;
- to send the queries results on keywords;
- a user interests sending;
- the agent search removing.

Personal agent interacts with the source agent to perform the following task: to obtain information about data sets loaded by user.

Personal agents interact with each other to transfer the information about users of the system.

User agent responds to events described in Table 6.4 to interact with the agents of the system.

<table>
<thead>
<tr>
<th>Name developments</th>
<th>Type developments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_search_history</td>
<td>receive</td>
<td>The user history's lines queries return. The search agent initializes this message</td>
</tr>
<tr>
<td>get_search_history_results</td>
<td>receive</td>
<td>Full queries history return. The search agent initializes this message</td>
</tr>
<tr>
<td>get_interests</td>
<td>receive</td>
<td>The user queries return. The search agent or source agent initializes this message</td>
</tr>
<tr>
<td>add_to_search_history</td>
<td>receive</td>
<td>Addition of information about user query. The search agent initializes this message</td>
</tr>
<tr>
<td>add_search_history_result</td>
<td>receive</td>
<td>Addition of information about user query results. The search agent initializes this message</td>
</tr>
<tr>
<td>change_user_type</td>
<td>send</td>
<td>The query to change user type. The source agent initializes this message</td>
</tr>
<tr>
<td>get_user_info_by_interest</td>
<td>receive</td>
<td>The user accordance to the interests return. The user agent initializes this message</td>
</tr>
<tr>
<td>add_user_download_dataset</td>
<td>receive</td>
<td>The conservation of loaded dataset. The source agent initializes this message</td>
</tr>
<tr>
<td>add_user_upload_dataset</td>
<td>receive</td>
<td>The conservation of dataset loaded into the system. The source agent initializes this message</td>
</tr>
</tbody>
</table>

Table 6.4: The agent user events
The diagram of user agent classes is shown in figure 6.10.

All the goals of the agent are designed to plans. Each plan is the appropriate class with at least one “body” method. The LoadUserPlan is performed while user authentication. The LoadUserPlan loads all the data about the user from the database, the history of user queries and results to them. To view information about the user the following sequence of actions occurs in the system:

- the user loads the profile view (Figure A.5);
- system invokes a Web service to work with user agent;
- Web Service initiates an agent goal, which is to update the user information;
- to fulfill the goal the agent fulfills a plan;
- the results are returned to the Web service, as a result of goal achieving;
- Web service returns the corresponding result to the Web interface.

![User Agent Class Diagram](image)

Figure 6.10: The user agent class diagram

To update information about user the following sequence of actions occurs in the system:

- the user enters the updated information into the system (Figure A.6);
- the system invokes a Web Service to work with user agent;
- Web Service initiates an agent goal, which is to get the user information;
- to fulfill the goal the agent fulfills a plan;
- the results are returned to the Web service, as a result of achieving the goal;

CommunicatePlan and CommunicateResponsePlan Plans are used to search users with relevant interests. Arrangement of these plans as follows:

- user requests an information;
- user agent initiates a CommunicatePlan plan which interrogates all user agents in the system;
- in the survey, CommunicateResponsePlan plan is initiated at each user agent, which checks whether the interests of its user correspond to the user interests requested and returns the result to requesting agent;
- agent returns the information to the Web service.

SaveUserDownloadPlan and SaveUserUploadPlan Plans are performed when a personal agent receives a request from the source agent. These plans are increase the rate of datasets loading and discharging by user from the system.
6.7.3 Coordinator [Manager] agent Scenario

To manage the overall system, registration and authorization of users in a “data and metadata exchange repository” operates the manager agent. The manager agent always suspends user and other agent’s queries. Agent Manager exists in the system as a single copy. Agent Manager is parallelized by agent platform. Manager Agent provides functionality from the standpoint of the user schematically shown in Figure 6.11.

![Figure 6.11: The use case diagram for coordinator agent](image)

The user sets the following tasks for the agent manager:
- user registration in the system;
- to obtain information about universities;
- user authentication and authorization in the system;
- changing the user status;
- to provide the user with administrator privileges;
- to obtain information about user activity in the system;
- information about users of the system obtaining, user deleting.

There are two main types of users in the system: authenticated and not authenticated.

The main difference between these two types of users is in that all not authenticated users have limited opportunities. They work with permanent agents only. A custom user agent is created for each authenticated user and also an additional search agent is created after the search query. Authenticated users are divided into the following groups: beginner, advanced user and administrator.

Manager Agent stores the following internal information about the current state of the system:
- the number of users, who use the system;
- the number of beginners, who use the system;
- the number of advanced users, who use the system;
Manager agent modifies its internal data considering the type of user when user authentication was specified. Table 6.5 shows the manager agent knowledge and its type.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>login_users</td>
<td>Belief, Integer</td>
<td>the number of users, who use the system</td>
</tr>
<tr>
<td>beginners_online</td>
<td>Belief, Integer</td>
<td>the number of beginners, who use the system</td>
</tr>
<tr>
<td>experienced_online</td>
<td>Belief, Integer</td>
<td>the number of advanced users, who use the system</td>
</tr>
<tr>
<td>admins_online</td>
<td>Belief, Integer</td>
<td>the number of administrators, who use the system</td>
</tr>
<tr>
<td>generalData</td>
<td>Belief, Object</td>
<td>general information about the universities of the system, countries, cities</td>
</tr>
<tr>
<td>methods</td>
<td>Belief, Object</td>
<td>the list of methods which available in the system</td>
</tr>
</tbody>
</table>

**Table 6.5: Coordinator agent knowledge**

Manager Agent receives information from the end users and from the environment. Input information from end-user of the system:
- account and password;
- user registration data;
- user account that needs to be transferred into the status of the administrator;
- user account that to be deleted.

End users interact with the agent manager. They invoke a Web service manager from the user interface.

System Administrators remove the users from the system by sending a request to the manager agent. For user to obtain the administrator rights, the other user-administrator should send the request to create a new administrator account on the basis of an existing one. Input information from the user agent: user account whose status should be changed. The transition from the one status to another is carried out by manager agent at the request of user agent of specific user. The Algorithm for the transition as follows: the user agent monitors the user activity in the system, and after getting some experience in the system, the agent prompts the user to raise his status and to receive additional options. If the user agrees the agent sends a request to the manager agent to change the type of user. Also the user invites to enter additional information about himself to obtain additional options. Manager Agent sends the information to other agents and transmits the information to the end user through a Web service. Output information for the end user:
- list of universities in the system;
- list of countries;
- list of cities;
- registration result;
- authentication result;
- the result of the transition to the administrator status;
- the user removing result;
- personal data of the system users;
- the number of users, who use the system;
- the number of beginners, who use the system;
- the number of advanced users, who use the system;
- the number of administrators, who use the system.
Registering a new user the manager agent checks the uniqueness of user name, and if successful, stores the user in the system.

Output information for the user agent:
- user agent creation;
- user status changing result.

Manager Agent interacts with the user agent to perform the following tasks:
- user agent creation;
- user status changing.

To interact with the system agents the agent manager has the following events described in Table 6.6.

<table>
<thead>
<tr>
<th>Name of event</th>
<th>Type of event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>change_user_type</td>
<td>receive</td>
<td>User status changing. The user agent initializes this message when the user agent want to change user status.</td>
</tr>
<tr>
<td>recalculate_raiting</td>
<td>send</td>
<td>The message sending from source agent to user after transferring him to a new status.</td>
</tr>
</tbody>
</table>

Table 6.6: The coordinator agent events

The diagram of manager agent classes is shown in Figure 6.12.

In LoginPlan the sequence of actions as following: set the parameters of the plan goals: the login and password. When you start the plan these options are used to build a query to the user database. If the user was not found or the password does not conformable to the password in the system, so the plan sets the corresponding result in the output parameter. If a user was found, so user ontology model is added to the agent presentation for further use.

![Diagram of Manager Agent classes](image)

Figure 6.12: The diagram of Manager Agent classes

To log the user in the system the following sequence actions occurs:
- user enters a username and password into the system (Figure A.1);
- system invokes a Web service to work with the manager agent;
- Web Service Agent initiates a goal of agent, which is to authenticate the user;
- to fulfill the goal the agent fulfills the plan;
- results are returned to the Web service as a result of goal achieving;
Web service returns the corresponding result to the Web interface.

To register a user in the system there are two options: Log beginner and advanced user registration. The administrator registration is available in both cases because the rights to administrator must be given by the other administrators.

The sequence of actions in registering as following:

- user selects the type of registration (advanced user, beginner);
- the beginner fills in his personal data (Figure A.2);
- the advanced user also fills in information about his interests (Figure A.3);
- quotations system invokes a Web service to work with the manager agent;
- Web Service Agent initiates a goal, which is to register the user;
- to fulfill the goal the agent fulfills the plan;
- results are returned to the Web service, as a result of goal achieving;
- Web service returns the corresponding result to the Web interface.

GetUsersInfoPlan and GetMethodsInfoPlan Plans are performed by starting the system in full that is constructed query to the database. It withdraws data about the methods of study represented in the system about the university, which researchers involve in the project, the cities and countries of universities location. After downloading the agent stores this information in its knowledge and all subsequent invocations of the goals, after these data obtaining, return these data from the knowledge.

Information about the methods is available at the home page (Figure A.4). SetupAdminUserPlan and DeleteUserPlan Plans are performed with the relevant user requests to Web services. They can be initiated only by a user, who has administrator rights.

### 6.7.4 Source agent Scenario

The main functions of the source agent are:

- scientific data sets addition;
- interaction with the user agent to display the newly added samples to the user depending on the user's interests. The Source agent informs the user agent about adding of scientific datasets to show users information about it after adding a new set to the storage;
- Metadata of datasets edit. The users, who create system or administrator have the possibility to edit;
- metadata dataset extract from repository;
- selection of entire information about a specific dataset and detailed information may be viewed only by registered users;
- to establish the dataset status numbers of downloads depending on estimates. The rating can be mark to each dataset. The rating assigned using the professional coefficient of a user, who makes it. At the moment of assess its assessment multiplied by a coefficient. This function performs source agent. The source agent should request the user agent ratio, calculate the result and save it in the database. Status of sampling can also increase depending on the number of downloads;
- interaction with the user agent to modify the coefficient of user professionalism depending on the status of scientific data sets, which he has added to the assessment or in storage;
- datasets filtering of metadata datasets by a specific parameter;
- new datasets adding to the repository that were found by search agent in the Internet.

Let us consider the diagram of sequences of a new dataset adding. Registered users should go to the page Create New Dataset. Fill in all required fields, click Insert button, Web page will call the service source to add a new set of statistical data to the repository (Figure 6.13).
The Service invokes the source agent for adding new dataset to the repository. After the agent added data to the database the agent invokes the manager agent to find all users with preferences, which correspond to just added added dataset.

![Diagram](image)

*Figure 6.13: The sequences diagram - new dataset adding to the repository*

Figure 6.14 shows the diagram of sequences. It reflects the process of datasets request from repository. Certainly with the huge number of datasets the request will be done long enough, so accordingly the page rendering will be long too. Therefore to optimize the process we need to use both page paging and request paging. This is a good practice of the professional systems.

![Diagram](image)

*Figure 6.14: The sequences diagram review of all repository datasets*

Figure 6.15 shows the diagram of sequences to estimate some statistical dataset. The user of the system estimates on the dataset page and it invokes service source. Service invokes source agent. Assessment plan is performed. During the plan source agent invokes profile agent. It returns the professionalism coefficient by which the assessment exposed. The source agent calculates the product of the professionalism coefficient and assessment and enters the result to the database.
All other source agent functions have the same realization.

The diagram of Source Agent classes is shown in Figure 6.16.

As we told early all agent goals are projected to plan. Each plan is the appropriate class with at least one “body” method.
6.7.5 Classification Scenario
We have an opportunity text (document) classification that came into our system with incomplete or missing set of information about authors, etc. and include it in a file folder for a specific category, i.e. a user does not complete information, but our system is able to attribute it to the appropriate category.

6.8 Agents development using Jadex technology
The system must know the properties of the agent to create and run the agent. The state of the agent is determined by beliefs, goals, current plans, as well as libraries of known plans. Jadex uses the declarative and procedural approaches for implementing the components of the agent. The body of the plan is executed as ordinary Java classes. All other notions (beliefs, goals, filters, and conditions) are defined by language. They are allowed to create Jadex objects in a declarative manner. The program developer can refer to the Java code, for example, to define methods. Full identification of the agent is reflected in the so-called agent definition file (ADF). In the ADF file the developer defines the initial beliefs and goals, announcing Java facilities. Announce plans to show the necessary classes from Java code. In addition to the BDI components in ADF file can be stored, some other information, for example, the default arguments for starting the agent or service descriptions for the registration of the agent in the facilitator directory. The structure consists of Jadex API, performed by the model, reusable common features. API provides access to the concept Jadex during programming plans. Plans are obvious classes Java. It is extend a special abstract class which provides a useful method of sending messages, the organization of secondary objectives or expectations of the events. Plans are able to read and modify the agent's thoughts. It uses the API framework agreement. Special function Jadex is that, in addition to the direct extraction of the remaining facts, intuitive OQL - like query language is allow to formulate a random complex expressions using the facilities which are contained in the database views. In addition to plans, coded in Java, provides the developer based on the XML agent definition files (ADF). It establishes the initial thoughts, objectives and plans of the agent. The Jadex mechanism reads file and starts the agent. It tracks its goals during a continuous selection of steps and launches a plan based on internal events and messages from other agents. Jadex is equipped with some advance features - such as access to the directory facilitator service. Feature encoded in the individual plans, linked agent used in many modules which are called abilities. Ability is described in a format similar to the ADF. It can be easily incorporated into existing agents. So summarize, in Jadex agents is thought, can be any type JAVA-site and stored in the database views. Objectives - explicit or imply descriptions of conditions that must be achieved. The agent executes the plans to achieve their goals. They are JAVA code procedural means.
7 System program model: Deployment and Implementation

This chapter presents the overall structure of the system. It describes all levels of the system.

The Java Web Data Mining Repository (ontology data and metadata exchange repository) was developed to support information research and development of Data Mining contextual use (Allinson, et al., 2008). The idea of data and metadata exchange repository is a system that will unite people around a favorite affair, occupation, hobbies, will allow sharing ideas, giving and receiving advice, recommendations in Data Mining and statistical research. This is the realization of well-known idea about Web 3.0 as social reference institutions based on the principle of automatic recommendations. According to experts (O'Reilly, et al., 2007) this system will differ from those of Web 2.0, users not only create the content, but also certify it: they marked that what deserve attention of those, who holding the same views. System allows to do this automatically. For example, based on user preferences stored within the system, the user of this system issued a list of recommendations - those whose interests closest coincide with yours.

7.1 Problem-solving

These problems could be solved in that case only if to define that the best transmitter of knowledge for a person (not just «interesting information», namely the urgent knowledge) is the other person, not a robot. Provided, that this person is not an amateur, but expert in this field of knowledge. In this case, the method of Web 1.0 to search the necessary statistical data would meant to place their questions to scientific sites or an independent search of foreign articles with data. Method is inconvenient: there are lots of parameters and through the Internet you can check only a few simplest ones, to check the most important parameters impossible at all. On the other hand the Web 2.0 may be used i.e., interviewing the acquaintances through communities or social networks. Here will be in other way: a lot of sympathetic and experienced people, but none of them, unfortunately, has any necessary data sets. That method implemented in the practical realization in this work can be well characterized as Web 3.0. That suggests to researchers, teachers, students and other categories of interested users to download their statistical datasets including their metadescription, to view already downloaded ones and to give advice to others regarding this or other dataset.

Such users work is human strategy of «manager of knowledge», which leads the users to the desired results. The conventional technique to organize the process of information searching in databases provides for personal request of user via Internet to Data Mining Repository server with request a summary of the responses result and its treatment. Performance, in general, of routine operations may take the experts a lot of time. In this regard becomes acute the problem of development of multi-agent system to automate the process of the queries in the information system, which has to assume much of the routine operations of information in database systems. The overall structure of the system can be represented in Figure 7.1.
The system consists of the presentation level, service level, agent’s level and database.

7.2 Presentation level
All the Web part (Web pages) is a presentation level of the system. Presentation level was built using html-pages and Wicket Framework. Wicket is open software based on Web components. The pages divide into Markup files and code. Code is written on the Java language, an excellent support for localization and styles to pages, no xml-file configuration, easy integration with Java security. . Net programmers can easily compare it with ASP.NET pages. Of course now there are many frameworks for developing web applications but most frameworks have weaknesses in supporting the state of server components page. Wicket makes this support easy and transparent. Wicket operates independently as server components pages. Programmers do not need to personally use the Http Session object wrappers or similar storage condition. This is one of the of Wicket goals. Wicket pages scheme is shown in Figure 7.2.
One of the parts of the general data and metadata exchange repository system was to create web-pages using the program container Tomcat 5.5.

The Wicket page CreateDatasetPage.html of the system is shown in Figure 7.3.

![Figure 7.3: Markup CreateDatasetPage](image)

It is provided that using the multi-agent developed client part, the user forms a query to a distributed information system. This request passes through all the system levels to the server database. The sequences diagram which work with the dataset of the system part is shown in Figure 7.4.

![Figure 7.4: The use case diagram](image)
In this section describes dataset part of the system. It may be considered the page to download data sets in the system, review of all data sets, view of detailed information about the datasets, edit metadata datasets. Figure 7.5 shows diagram of presentation level class and service resource class.

The package ua.kture.dmr.common.beans.dataset have classes DataSet, DataSetFile, Judge. Each is an objective representation of metadata ontology resource. All classes of the system operate the objects of these classes. Package ua.kture.dmr.agents.dataset provides classes plans agents:
- InsertDatasetPlan - performs plan insert_dataset, adds a new dataset to the repository;
- ReadAllDatasetPlan - performs plan read_all_dataset, reads all data samples from the repository;
- ReadDatasetsBySlotPlan - performs plan read_dataset_by_slot, reads all data samples that match the query from the repository;
- AppraisalDatasetPlan - performs plan appraisal_dataset, adds a specific set of assessment data, adds comment data set;
- UpdateDatasetPlan - performs plan update_dataset, obnovlyuye data samples;
- InsertDatasetFile - performs plan insert_dataset_file, adds the files of statistical data sets.

Let us consider one of the plans. The sequence of actions in ReadDatasetPlan as following: from the objectives is set the option plan «dataset name». When you launch the plan, these options are used to construct the query to the database resource. If the dataset was not found, the plan establishes the corresponding result in the output parameter. If dataset was found, the representation of the agent added the ontological model for its further use. To transfer data between the server and ResourceAgent is used the network connection - socket. Sockets interface is able to transmit data between two applications that work on the same or different nodes of the network. Socket is created as an object of Socket class, specifying the server host and port number used by the server. server_socket = new Socket (server_name, server_port);
Here are the input and output streams for the exchange of information. On the client side, the operation is performed in the same way as on the server side. server_receive = new BufferedReader (new InputStreamReader (server_socket.getInputStream ()));
server_send = new PrintStream (server_socket.getOutputStream ());
Upon successful connection ResourceAgent server transmits data and sql-query command. This command means. If the server data transfer occurs normally, the agent informs the user that his request accepted by the system for processing, i.e, gives the result. It completes its work, closing network connection.
finally (if (server_receive! = null) server_receive.close ();
if (server_send! = null) server_send. close ();
if (server_socket! = null) server_socket. close ();

As already noted, the multi-server part of system is implemented in Java. This ResourceAgent provides communication interface with other agents and repository server system. This four-level repository architecture provides the opportunity to interact with developed repositories of services that are very urgent practice now and at the level of agents i.e. the interaction between agents of different systems is possible.

Package ua.kture.dmr.jwsx.ui.pages contains AbstractPage class, which is basic to all pages of the system. It creates a menu for each site page.

Package ua.kture.dmr.jwsx.ui.pages.dataset contains website pages CreateDatasetPage, DataSetListPage, DataSetDetailsPage, UpdatedataSetPage, which are inherited from AbstractPage base class. This package provides an interface to the data sets.
Package ua.kture.dmr.jwsx.wsimpl allows ResourceServiceImpl class provides implementation for queries to the source agent. ResourceServiceImpl class inherits from the AbstractAgentWebService class, which is the base for all system services and realized the ResourceService interface.

Features of class are listed below:

- `insertDataSet (DataSet dataset) throws Exception;`
- `getDataSet (SessionInfo sessionInfo, String title) throws Exception;`
- `getAllDataSets (SessionInfo sessionInfo) throws Exception;`
- `getDataSetsBySlot (SessionInfo sessionInfo, String slotName, String slotValue) throws Exception;`
- `insertDataSetFile (DataSetFile datasetFile) throws Exception;`
- `setDataSetMarkComment (Judge judge) throws Exception;`
- `updateDataSet (DataSet dataset) throws Exception;`

7.3 Service level

Data Mining Repository is a service-oriented architecture that meets the principles of multiple usages of the functional elements, eliminate duplication of functionality in the software, unification typical operating processes to ensure the operating model of centralized processes and functional organization based on the industrial platform integration. Components of the program can be distributed on different nodes of the network and offered as independent, weakly connected, which can follow service applications.

A developed software system is implemented as a set of Web services. It integrates using SOAP and WSDL. Interface of program components provides encapsulation of implementation details of specific component from other components. Thus, this architecture provides a flexible and elegant way to combine and reuse of components. In order to process the requests from users with Web interface was developed a multi-system. Web page requests directed to the web services system, which in their turn send requests to agents. The system supports four Web services:

- Administration Service;
- Search Service;
- Profile Service;
- Source Service.

To develop web services was implemented the xfire solution. Xfire is a free solution that
solves the problem of interoperability, implementation of various problems of industrial standards. Developers of distributed applications this is the easiest mechanism for implementing of remote requests. In this part of the work we describe Resource Service. ResourceService service has a method that fully covers the functionality of the source agent.

- insertDataSet (DataSet dataset) throws Exception;
- getDataSet (SessionInfo sessionInfo, String title) throws Exception;
- getAllDataSets (SessionInfo sessionInfo) throws Exception;
- getDataSetBySlot (SessionInfo sessionInfo, String slotName, String slotValue) throws Exception;
- insertDataSetFile (DataSetFile datasetFile) throws Exception;
- quotations setDataSetMarkComment (Judge judge) throws Exception;
- updateDataSet (DataSet dataset) throws Exception.

7.4 Agent subsystem
In the system of data and metadata exchange repository (Data Mining Repository) all agents, which multi-system includes, belong to one of the following types:
- manager agent, running on the server and coordinates the work of users;
- user agent that performs the interaction with users;
- resource agent, responsible for datasets operations;
- agent search, performing the information search.

Thus, even if agents are placing on different servers, it will be possible to interact with queries from users. The multi-server system includes agents ManagerAgent, ProfileAgent, ResourceAgent, SearchAgent (each agent was described in details in section 6). Messaging between agents based on the HTTP protocol and work with the database is via JDBC one.

7.5 Data and metadata exchange repository explanation
Compare sections 7.1, 7.2, 7.3 with the overall system structure presented in the figure 7.1 we received detail explanation of the data and metadata exchange repository, which is presented on figure 7.6.

Figure 7.6: The explanation of overall system
7.6 Work database level
To store ontological models there was used the database management system Oracle 10g, namely a new option Oracle Spatial (Oracle, 2005). Each ontological model is designed for RDF DATA MODEL in Oracle Spatial. Thus we get three models:
- Users;
- Datasets;
- Methods.

Figure 7.6 shows a model to store RDF statements in Oracle Spatial 10g.

<table>
<thead>
<tr>
<th>PREDICATE</th>
<th>PREDICATE</th>
<th>OBJECT</th>
<th>OBJECTS$R</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a></td>
<td>URI</td>
<td><a href="http://dmr.kture.ua/dataset/DataSet">http://dmr.kture.ua/dataset/DataSet</a></td>
<td>LIT</td>
</tr>
<tr>
<td><a href="http://dmr.kture.ua/dataset#Title">http://dmr.kture.ua/dataset#Title</a></td>
<td>URI</td>
<td>CMU_Face_Images</td>
<td>LIT</td>
</tr>
<tr>
<td><a href="http://dmr.kture.ua/dataset#DataSetAbstract">http://dmr.kture.ua/dataset#DataSetAbstract</a></td>
<td>URI</td>
<td>This data consists of 640 black and white face images of people taken with varying pose (straight, left, right, up), expression (neutral, happy, sad, angry), eyes (wearing sunglasses or not), and size</td>
<td>LIT</td>
</tr>
<tr>
<td><a href="http://dmr.kture.ua/dataset#AnalysisMethod">http://dmr.kture.ua/dataset#AnalysisMethod</a></td>
<td>URI</td>
<td>Classification</td>
<td>LIT</td>
</tr>
<tr>
<td><a href="http://dmr.kture.ua/dataset#AttributeAmount">http://dmr.kture.ua/dataset#AttributeAmount</a></td>
<td>URI</td>
<td>0</td>
<td>LIT</td>
</tr>
<tr>
<td><a href="http://dmr.kture.ua/dataset#AttributeInfo">http://dmr.kture.ua/dataset#AttributeInfo</a></td>
<td>URI</td>
<td>N/A</td>
<td>LIT</td>
</tr>
<tr>
<td><a href="http://dmr.kture.ua/dataset#AttributeType">http://dmr.kture.ua/dataset#AttributeType</a></td>
<td>URI</td>
<td>Integer</td>
<td>LIT</td>
</tr>
<tr>
<td><a href="http://dmr.kture.ua/dataset#DataType">http://dmr.kture.ua/dataset#DataType</a></td>
<td>URI</td>
<td>Image</td>
<td>LIT</td>
</tr>
</tbody>
</table>

Figure 7.7: A model to store RDF statements in Oracle Spatial 10g

DBMS Oracle Database 10g was the first large-scale project to implement storing ontologies in spatial form. Oracle Spatial is DBMS Oracle Database 10g technology which includes additional features for handling spatial data to support spatial services, various programs for processing or to provide information on the location of objects and other information systems. DBMS support includes Oracle 10g RDF / RDFS, allowing developers to use the platform to take advantage of semantic data. Application developers can add value to data and metadata, defining new sets of conditions and relations between them. This set of terms (ontology) is more suitable for query and analysis based on the semantic approach than conventional datasets. Ontology datasets often contain millions of data elements and relations between them. Its can be grouped in triplets using the new RDF data model. Oracle admits triplets billion expansion to meet the requirements of most applications.

How to store RDF in Oracle Spatial 10g:
- RDF data is stored as directed, logical graph;
- Subjects and objects are displayed as nodes and predicates as relations, in which the subject is an initial node and final is object;
- Relationships are a complete RDF triplets;
- Oracle Spatial RDF data model;
- RDF data model supports three types of database objects: a model (RDF graph consisting of a set of triplets), base of rules (set of rules), the index rules (aimed RDF graph). To implement the semantic query is used SDO_RDF_MATCH operator;
The main advantages of Oracle Spatial 10g are:
- Support for decentralized data management;
- Support of all RDF data types;
- SQL search and recovery of RDF models;
- Making queries to the RDF model, using the circuit graph;
- A query RDF (SPARQL) with other operators in SQL;
- A logical conclusion based on RDFS (RDF schema) rules;
- The logical conclusion based on policies defined by the annex.

RDF Model is stored as a graph: nodes - URI objects, certain set of links between nodes, W3C RDF Schema recommendation describes the dictionary, is applied to describe other dictionaries.

RDF documents are stored as a triplet (subject, property, and object) and use the reduction to represent namespace. Triplet is used to store table MDSYS.RDF_VALUE $. Maintain custom system of rules of inference. Rule consists of: the terms «if», «filter», «so».

The SDO_RDF_TRIPLE (subject VARCHAR2 (4000) type, VARCHAR2 (4000) property, VARCHAR2 (10000)) object are used to display the triplets. SDO_RDF_TRIPLE_S is a type to store triplets actually refers to the data in the table of model. A free library of Jena 2.0 is used for interaction with the database agents.
8 Conclusions and future challenges

The main topic of this thesis focuses around one fundamental principle extracted from ontologies and intellectual agents. Thus, in this chapter we discuss the application, concerning the implementation of “Data and Metadata exchange repository” and the results that have been attained. We present the implication for the future work of our repository.

8.1 Results
The results of master thesis work research is developed multi-agent system for processing and storage of any statistical data. Research of existing repositories allowed identifying the main bottlenecks of the similar statistical repositories that were taken into account. Operating with UCI repository the user is able to filter, according to subsection of data mining area, the data files to view the brief characteristic of a file, to download a file. Using DEA Dataset Repository the user is able to search in any of criteria, view the brief characteristic of a file and to download a file, but only after .XML registration. In Data Repository the user can download any file of subjects without registration. Operating with Frequent Itemset Mining Dataset Repository the user does not need to be registered, he can obtain the information about researches made on samplings and the contact information of researchers, to download a file.

A key feature of the developed system via above mentioned typical statistical repositories is implementation of the datasets metadescription using the European standard SDMX 2.0 and ontological models that are stored in the system.

The advantage and novelty of the work is implementation a set of the ontological models of Data mining methods, which is used for the selection of a proper method under the sample source of the user datasets. To work with set of the ontological models have been developed a set of search algorithms that implement simple and advanced search supporting, account search, which takes individual interests, orientation of activities, previous search queries of the user, as well as architecture of search module based on these search algorithms. Also this system (intelligent data and metadata exchange repository) has a taxonomy of DM methods that allows to establish connection between DM methods and data on which they can be applied, that for the user of "beginner" class represents itself as the expert system. The user ontological model, resources ontological model have been developed in protégé version 3.4, which allows working fast with ontologies.

For ontological models interaction and implementation of search algorithms it was developed a set of general intelligent agents models. They can be used as a mechanism for displaying information on the ontological models, as well as a mechanism for user interaction with the system. This set of general models include model for integrating intelligent agents with web systems, a model of intelligent search agent, and model for relationship between agents. The user of the developed intelligent data and metadata exchange repository is able to make formal description of the user’s problem domain (filling in the necessary fields in the ontology model) and formal description of the dataset which is need for specific tasks. All this kind of activities is a part of the search agent. The search agent, having processed the received information, transfers it to the coordinator agent and via the search agent the necessary connection with a data file is made. The user intelligent agent (user agent (profile agent)) allows to personalize the answer to the following questions: what is the user name; what is e-mail address; what language the user prefers; what are current goals of the user; whether the user is beginner or advanced one; what academic institution the user belongs to; what are localization preference of the user; whether the interests of user coincide with other users interests in the system; what are recent inquiries of the user. The result of applying the multi-agent approach for creating such system is the ability to perform a simple search for users regardless of user type; to search by different criteria for authorized users; to provide popular data sets; to perform a search taking into account the personal needs of the user; to provide
user relevant queries information; to keep statistics of requests and, if necessary, provide this information; to remember the successful search results.

Here is used the cross platform programming language Java, multi-agent platform Jadex, database server Oracle Spatial 10g, and also the development environment for ontological models – Protégé Version 3.4. Database management system Oracle Spatial 10g which allows to work with ontologies in RDF format was chosen as a method of resource ontological models storage. Development environment of ontological models is Protégé.

8.2 Conclusions
Currently, there are many repositories of scientific datasets. The main disadvantages occurred in these systems are: text-only format is not convenient to use and to change the format of files, not user-friendly interface, and the search is only by one of many criteria, i.e. not allowed to combine the search for a number of conditions, poor search.

In many systems, there is no any understanding for what tasks you can use this dataset, there is also insufficient information on the data. Currently, the agent technologies are widespread, where the main part is the agent - a software entity capable of such qualities as autonomy, activity, commitment, mobility, sociability. The creation of ontologies is a prospective direction of up-to-date research in processing of information provided in natural language. One of the advantages of using ontologies as a tool for learning is a systematic approach to the study of the subject area. Meanwhile achieved: regularity - Ontology provides a holistic view of the subject area, uniformity - the material presented in a unified format is much better perceived and reproduced; scientific - Building the ontology allows to restore the missing logical link in their entirety. Also, ontologies allow the use the great volumes of data from different systems, due to the fact they creating the semantic description of data. a) studied the main stages of work with the repository of scientific research data sets; b) reviewed the existing repositories of scientific data sets, to identify their strengths and weaknesses; c) studied the technology Semantic Web; d) investigated the possibility of agent technology; e) analyzed the ways to develop a web-oriented multi-applications; f) developed the architecture of multi-repository of scientific data sets; g) developed the ontological model of the user; h) developed and realized as a software BDI agent model of the user; i) developed and realized as a software BDI agent model.

8.3 Future challenges
The Data and Metadata Exchange Repository (Data Mining Repository) is a complete software product but has many ideas that will be implemented in the future.

Nowadays a good practice is to put into effect a new idea, even if it is not fully realized, but by what lure the users. The practice development has several stages. The most frequently mentioned short classification of stages of development, according to which the system passes 5 stages in its development: seed stage, startup stage, growth stage, expansion stage and exit stage.

In further stages of development of our repository ideas are implementing such functionality:
- advanced search sets from various sources on the Internet, using an algorithm for clustering analysis of data for only that of sample data on a given dictionary of terms;
- conversion of various file types;
- generation of samples for specific formulas for simple images;
- expand the idea of recommendations.

Further this system can be improved by the development of its ontology, and an increase in the number of agents. Agents that could improve the system:
a) Pre-data agents, which would convert the sample into various formats;
b) Dataset checking agents, which would verify the correspondence of the data set by methods of research established for it;
c) The agent of data sets search in different repositories data sets, which could interact with the agent for pre-loading data to the data.

You can also add a subsystem of the articles and the results of scientific experiments, researchers conducted using data sets from the repository.

8.4 Reflections
This field has been selected for research proceeding from the practical needs of permanent use of statistical repositories. Last year during my baccalaureate work, where has been developed clusterization algorithm of linear-inseparable SATYR objects, I have faced the necessity of appropriate data file selection, namely intersected classes of various density. Having done enormous work on operation with various statistical repositories it was failed to find an appropriate file as there is no such information in the files description. I had to try successively all files of repositories manually on the filter clusterization and classification and then there was an idea of intelligent system development interacted with a repository, which would allow not only to store files and to filter them by the set inquiries, but also to create the formalized model of data files description, which is expanded indefinitely (that it is possible to carry out using the ontological approach only), the formalized model of the system user, search personification in a repository, that is possible only using the intelligent agents. Also to create the expanded system that is possible, only if the system is a models set and mechanisms to operate with them.

Ontologies creation is a direction of up-to-date research in processing of information provided in natural language. As the computer cannot understand as the person, a state of affairs in the world, representation of all information in formal shape is necessary for it. Thus, ontologies are for original model of world around, and they have such a structure, that they easily yield to machining and analysis. Ontologies provide the system with data on well described semantics of the set words and specify a hierarchical structure of area and interrelation of units.

The complete development of a repository will allow to solve the problem of data use for beginners, will allow all scientists to exchange the descriptive part of files in different application areas. Adding files by various scientists it will not be necessary to fill in formally all fields to add the files. It will be enough to give files description and the agent will automatically add it in appropriate section, and further will find it for user.
Glossary

KDD – Knowledge Discovery and Data Mining
DM – Data Mining
SDMX – Statistical Data and Metadata eXchange
MAS – Multi-Agent System
HTTP – Hypertext Transfer Protocol
SQL – Structured Query Language
XML – eXtensible Markup Language
HTTP – Hypertext Transfer Protocol
RDF – Resource Description Framework
RDFS – Resource Description Framework Schema
OWL – Web Ontology Language
URI – Universal Resource Identifier
SPARQL – SPARQL Protocol and RDF Query Language
BDI – Belief-Desire-Intention (software model)
ASCII – American Standard Code for Information Interchange
GZIP – GNU zip
W3C – World Wide Web Consortium
PRS – Procedural Reasoning System
ARS – Agent-Oriented System
dMARS – Distributed Multi-Agent Reasoning System
DBMS – Relational Database Management System
JDBC – The Java Database Connectivity
SOAP – Simple Object Access Protocol
WS – Web Services
WSDL – Web Services Description Language
ADF – XML based Agent Definition File
OQL – Object Query Language
References

W3C Recommendation (2004). Web Ontology Language (OWL) overview, viewed <http://www.w3.org/TR/owl-features/>, (090812)


Internet sites

http://www.kdnuggets.com, viewed 2009-08-12
http://www.w3.org/2002/ws/, viewed 2009-08-12
http://w3.msi.vxu.se/~wlo/files/WSWT06/Slides6.pdf, viewed 2009-08-12
http://jade.x.informatik.uni-hamburg.de/bin/view/About/Features, viewed 2009-08-12
http://wicket.apache.org/, viewed 2009-08-12
http://archive.ics.uci.edu/ml/about.html/, viewed 2009-08-12
http://iaastech.org/ias/reposit.htm, viewed 2009-08-12
http://archive.ics.uci.edu/ml/, viewed 2009-08-12
http://www.etm.pdx.edu/DEA/Dataset/default.html/, viewed 2009-08-12
http://www.cs.washington.edu/research/xmldatasets/, viewed 2009-08-12
http://fimi.cs.helsinki.fi/data/, viewed 2009-08-12
http://citeseer.ist.psu.edu/old/394923.html/, viewed 2009-08-12
http://en.wikipedia.org/wiki/BDI_software_agent/, viewed 2009-08-12
http://protege.stanford.edu/index.html, viewed 2009-08-12
http://shcherbak.net/sitemap/, viewed 2009-08-12
http://shcherbak.net/razrabotka-vysokoeffektivnyx-sredstv-sozdaniya-i-obrabotki-ontologicheskix-baz-znanii/, viewed 2009-08-12
http://dic.academic.ru/dic.nsf/ruwiki/611874/, viewed 2009-08-12
http://wicket.apache.org/, viewed 2009-08-12
http://jena.sourceforge.net/ontology/index.html, viewed 2009-08-12
http://www.magenta-technology.ru/technology/index.shtml/, viewed 2009-08-12
http://www.fipa.org/, viewed 2009-08-12
http://dic.academic.ru/dic.nsf/ruwiki/611874/, viewed 2009-08-12
Appendices

Appendix A  Data and Metadata Exchange Repository (Data Mining repository) [an example]

Figure A.1: The log and password page

Figure A.2: Beginner registration
Figure A.3: Advanced user registration

Data Mining methods hierarchy

- Data Mining Methods
  - Visualization
    - Diagram
    - Table
    - Correlation Matrix
    - BarGraf
    - Variance Diagram
    - Neural networks Count
  - Cluster Analysis
    - Nonhierarchical
      - EM
    - K-Means
      - K-modes
      - Bisection-K-means
      - K-medoids
  - Prediction
    - Regression Analysis
  - Association
    - Quantitative Association
    - Generalized Association
  - Classification
    - Neural Networks
    - Decision Trees
      - C4.5
      - CART

Figure A.4: Research methods information
Figure A.5: View user information

Figure A.6: The user refreshment
Appendix B  Data and Metadata Exchange repository (Data Mining Repository)

Figure B.1: Simple search page

Figure B.2: Advanced search page
Figure B.3: Search results page

Figure B.4: The prompt-page of the system of the popular queries
Figure B.5: The list of the most popular datasets of repository on the home page