BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **12**, No 4

Sofia • 2012

SINUS – A Semantic Technology Enhanced Environment for Learning in Humanities

Gennady Agre

Institute of Information and Communication Technologies, 1113 Sofia
Email: agre@iinf.bas.bg

Abstract: *The paper describes a semantic technology based environment intended for developing technology enhanced learning applications in humanitarian problem domains. The environment consists of three layers: the storage layer contains heterogeneous repositories storing domain and pedagogical knowledge; the tool level contains a set of tools for processing different types of knowledge and the middleware layer is implemented as an extended search engine carrying out all necessary communications between the tools and the repositories. Some implementation issues are discussed and a preliminary evaluation of the environment based on an exploitation is presented.*

Keywords: *Technology enhanced learning, semantic technologies, service-oriented architectures.*

1. SINUS project – objectives and goals

The main objective of the ongoing national funded research project SINUS: Semantic Technologies for Web Services and Technology Enhanced Learning (sinus.iinf.bas.bg) is to provide a semantic technology based environment allowing developing Technology Enhanced Learning (TEL) applications that use the existing heterogeneous software systems. The basic approach for realizing such applications is to exploit Service Oriented Architecture (SOA) able in a uniform way to use heterogeneous systems implemented as Web services. More precisely, the project is focused on:

- Developing an environment allowing in a uniform way querying, retrieving and using information objects stored in heterogeneous digital libraries. The environment should provide means for ontology-based upgrades of existing digital libraries without any changes in library structure, content and annotations.
- Extending the developed environment by specialized TEL-oriented tools implementing a active learning approach in problem domains characterized by intensive use of multimedia digital libraries. The tools should support the analytical skills of the learners in a given humanitarian discipline by authoring of analytical materials in some well defined learning situations.

The target groups of the SINUS environment cover academic users (students following different courses from the subject under study and their teachers), as well as non-academic users. The students from formal education forms are expected to have a middle or higher level of knowledge about the domain and to use the environment for improving their domain knowledge. They will actively search for digital learning resources and use them to achieve their learning goals by development of analytical scholarly essays, thematic projects, course or diploma theses, etc. The teachers should receive information support in preparing concrete learning tasks for their students (development of analytical materials for different purposes based on an appropriate selection of available digital resources), as well as in preparing exemplary learning resources and recommendations for students work on different levels [1].

The potential of such a specialized TEL environment should be illustrated by developing a prototype learning system assisting students in preparation of projects in the area of East Christian iconography.

The present article discusses some aspects related to the implementation and evaluation of SINUS environment. The next section briefly presents the environment conceptual architecture. Section 3 is devoted to implementation of the architecture – some main characteristics of its components are described. The fourth section is an evaluation of SINUS environment made on the project evaluation scenario – the developed approach for integrating heterogeneous repositories and the correspondence between project goals and achievements are discussed. The last section is a conclusion.

2. SINUS conceptual architecture

The SINUS environment consists of three layers – a storage layer containing heterogeneous repositories used for storing data and knowledge; a tool layer containing tools used for data and knowledge processing and a middleware layer used as a mediator between tools and repositories (Fig. 1).

2.1. The storage layer content

- *Basic digital library* represented as Web service. It is assumed to store (in a non-semantic manner) some “original” (i.e., created outside SINUS environment) annotations of multimedia objects (that can be stored inside or outside the library).

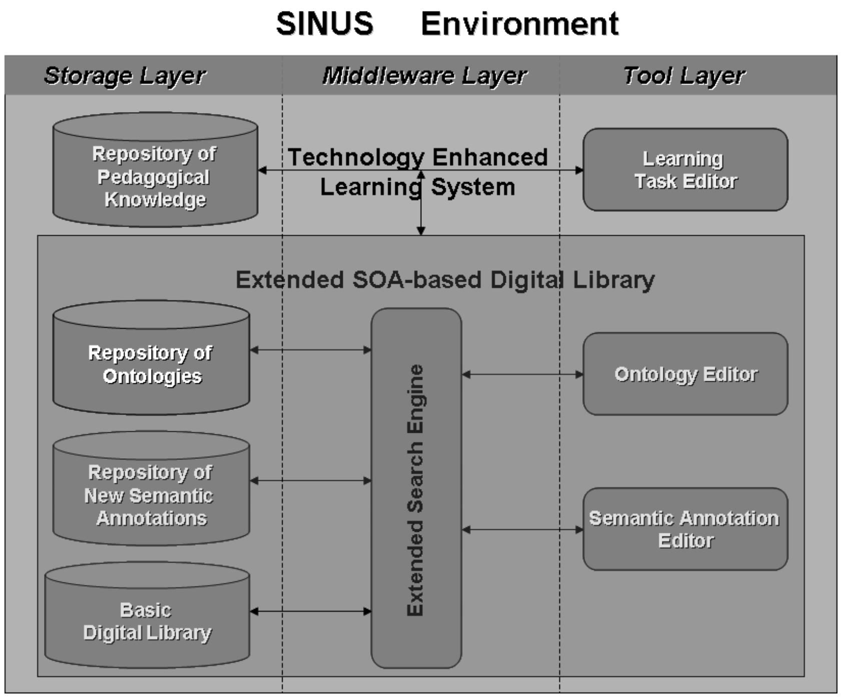
SINUS Environment
Storage Layer
Middleware Layer
Tool Layer
Repository of Pedagogical Knowledge
Technology Enhanced Learning System
Learning Task Editor
Extended SOA-based Digital Library
Repository of Ontologies
Repository of New Semantic Annotations
Basic Digital Library
Extended Search Engine
Ontology Editor
Semantic Annotation Editor

Fig. 1. SINUS conceptual architecture

- *Repository of ontologies* containing OWL ontologies describing basic and specialized domain knowledge. It is assumed that the basic ontology is an ontological extension of data scheme describing the basic digital library. The specialized ontologies describe some additional, more specialized concepts and relations of the domain. Such ontologies can be used for creating new specialized descriptions (semantic annotations) of objects from the domain including those already described in the basic library. The specialized ontologies contain also relations with some concepts from basic ontology and in such a way can be considered as some specialized extensions of the basic ontology.
- *Repository for new semantic annotations* of domain objects. Such specialized annotations represent the features of objects from the basic digital library that cannot be described in terms of basic ontology. They allow such objects to be queried and retrieved in terms of the specialized ontologies.
- *Repository for pedagogical knowledge*. The repository is intended for storing some formalized knowledge of teachers about learning tasks to be solved by students. In general such knowledge comprises descriptions of the learning task and some pedagogical recommendations (advices, strategies, etc.) how the task could be solved. In the case of SINUS besides textual descriptions of the tasks the knowledge includes also exemplary plans for finding task solutions, as well as one of the possible correct solutions found by the teacher him/herself. This knowledge is used for creating task models that can be applied for checking students' solutions for correctness.

2.2. The tool layer content

- *Ontology Editor* intended for creating and editing OWL ontologies. For these purposes each existing editor of OWL ontologies can be used. In SINUS project all ontologies have been created and edited by means of Protege – OWL [2].
- *Semantic Annotation Editor* – a tool for creating and editing semantic (RDF-based) annotations based on the specialized ontologies stored in the Ontology Repository.
- *Learning Task Editor* – a tool for creating and editing internal (SINUS) representations of learning tasks (task models). Such a representation includes textual description of the task, a possible plan for finding the proper solution, an example of such a solution found by the teacher, etc.

2.3. The middleware layer

- This layer is implemented as an *Extended Search Engine* – a complex Web service providing an effective way for accessing and retrieval of information stored in the heterogeneous repositories of SINUS environment. The engine receives queries (internally represented as SPARQL queries) sent by some of SINUS users (a human or a program) and processes them: after analyzing a query the engine converts it into a sequence of sub-queries corresponding to the type of SINUS repositories, in which the answer will be searched, sends the converted sub-queries to the corresponding repositories, processes and integrates the results and then returns the answer to the user. The extended functionality of the engine allows not only searching for objects satisfying the query, but creating new objects specifying in the query as well.

In the described-above SINUS environment a TEL application is built hierarchically starting with transforming an autonomous system for storing, accessing and retrieving multimedia data (digital library) into a Web service (basic library), then – into an extended digital library, which is based on Web services and ontologies, and finally – into a TEL system based on Web services

3. Realization of SINUS environment

3.1. SINUS heterogeneous repositories

SINUS environment has been constructed to work with repositories of different types but implemented as Web services. At the moment the environment uses three repositories – repository of basic objects and annotations (basic digital library), repository of new semantic annotations and repository of learning tasks.

According to SINUS exploitation scenario the “Virtual Library of East Christian Art” has been selected as a primary digital library [3]. This library is a fully autonomous and closed Web application implemented by means of relational database technology with built-in and hard-coded domain knowledge that, in practice, is a characteristic feature of most digital libraries with multimedia content. The library contains descriptions of different iconographical objects (icons, altars, etc.) represented according to a fixed hierarchical schema as most of the object features (e.g., description of iconographical scenes, restoration notes, etc.) are represented as unstructured texts. It also provides the user with a set of functions for searching the library and visualizing the search results. In order to be used in SINUS environment the library has been converted to a WSDL Web service preserving the whole functionality of the original library (except the function for updating the library content). Such a Web service is used in SINUS environment as the repository of basic annotations.

The second repository (RDF-based) is used for storing new semantic annotations. The term “new” means that such annotations are made in terms of new (i.e., specialized) ontologies, which have not been used for creating basic annotations stored in the basic library. The specialized annotations allowing searching for multimedia objects via querying their features that are neither presented in the annotation schema of the basic library nor can be inferred by a rule combining some of the basic object features. The Repository for new semantic annotations is implemented by means of a highly effective semantic repository OWLIM [4]. OWLIM uses Sesame[1] which is a de-facto standard framework for processing RDF data, as a library, taking advantage of its APIs for storage and querying, as well as the support of a wide variety of query languages (e.g., SPARQL and SeRQL) and RDF syntaxes (e.g., RDF/XML, N3, N-Triples, Turtle, etc.). OWLIM provides a possibility for effective reasoning based on such semantic languages as RDFS, OWL DLP and OWL Horst.

Since in SINUS environment the annotations of information objects are stored in different repositories, an effective mechanism for combined use of both types of annotations has been implemented. It allows searching and retrieving objects satisfying complex queries described in terms of both basic and specialized ontologies. The implementation of the mechanism is based on the following ideas:

- *Use of a specially designed mapping procedure for lifting basic annotations to the semantic (OWL) level* that guarantees uniqueness of the created identifiers for ontological class individuals [5]. As a result, the lifted basic annotations and the corresponding specialized annotations form a common virtual RDF graph describing an object both in terms of basic and specialized ontologies.
- *Use of an effective procedure for processing complex SPARQL queries* containing terms of both basic and specialized ontologies. If it is necessary the procedure dynamically lifts from the basic digital library some non-semantically represented annotations of objects that partially satisfy the query, writes the resulted temporary semantic annotations into the semantic repository, makes an inference for finding objects that fully satisfy the query and, finally, deletes all temporary semantic annotations from the repository. The procedure is described in more details in Section 3.3.

Since the procedure above described for searching answers on SPARQL queries requires a dynamic way for writing and deleting information to the semantic repository, we have used Version 1.1 of SPARQL query language [6] and

[1] http://www.openrdf.org/index.jsp

Version 4.2 of OWLIM that includes SPARQL 1.1 Update functionality, as well as SPARQL 1.1 query support.

Ontology repository is a conceptual component of SINUS environment aiming at storing domain knowledge represented as OWL ontologies. Although the ontology repository contains another type of data different from semantic annotations stored in the Repository for new semantic annotations, functionally both repositories are implemented as a single semantic repository OWLIM in order to exploit an effective search and inference engine built in the OWLIM.

A specific feature of SINUS environment is the use of ontologies for integrating data from different heterogeneous sources. All domain ontologies in SINUS are divided on basic and specialized ones according to their relations with external for the environment digital libraries. A basic ontology can be seen as a semantic explanation of information object descriptions stored in an external digital library relying on a relational database. Such a basic ontology converts and generalizes the database schema into an ontology model representing in an explicit way all domain knowledge built in the basic digital library. Basic ontologies allow querying information in a uniform semantic way (e.g., by means of SPARQL queries), no matter whether the requested information is stored in a semantic or non-semantic repository[2].

Specialized ontologies represent new domain knowledge that has not been used explicitly during construction of basic digital libraries. In SINUS we assume that a specialized domain ontology describes the domain from a particular point of view and can be seen as a specialized extension or development of some concepts from the basic ontologies where they are presented in a too general way. Thus some concepts (classes) from basic ontologies can be described in a more detailed way as their new additional properties or sub-classes could be presented in a corresponding specialized ontology. What is important is that all these extensions of domain knowledge allowing more rich and flexible access to objects from the basic libraries do not change the original structure and content of such libraries.

It should be mentioned that as basic as specialized ontologies can be used independently ones from others. That is why in order to provide a combined use of them it is necessary to define the correspondent links between the ontologies. In the general case this task for merging and alignment of ontologies is rather complicated [7]. In SINUS we have applied a pragmatic approach – some of the “root” concepts (classes) of a specialized ontology are declared as equivalent or sub-classes of the correspondent classes from the basic ontology. In such a way when these ontologies are used together, such basic concepts become extended by new sub-classes and/or new properties defined in the specialized ontologies.

SINUS environment is able to operate with numerous basic ontologies (each of which is corresponding to a separate basic digital library implemented as a Web service) and with several specialized ontologies. In the experimental application of the environment one basic and two specialized ontologies have been used. The

[2] Of course, a special procedure – mediator lowering a semantic query to the corresponding “language” understandable by a specific non-semantic digital library should be constructed. See for details Section 3.4.

basic ontology[3] represents an ontological extension of the metadata schema of the basic digital library “Virtual Library of East Christian Art”[3]. The ontology comprises 55 classes, 38 object properties and 28 data-type properties, where main classes are Iconographical Object with its sub-classes Icon, Wall-Painting, Miniature, Mosaic, Vitrage; Author, Iconographical Scene, Character, Iconographical Technique, Base Material and so on [8]. The first specialized ontology[4] describes some elements of technology used for creating iconography objects such as Gilding, Primer, Lacquering and so on [9]. It contains 16 classes, 14 object properties and 43 ontological individuals. The second specialized ontology[5] describes composition scenes of iconographic objects and comprises 7 classes, 5 object properties and 21 ontological individuals.

The third repository in SINUS environment is intended for storing knowledge related to educational aspects of the environment. At the moment it contains a formal model of a learning task “Creation of a dedicated multimedia collection”. In order to solve this task a student should intensively use the extended digital library of SINUS in order to find a set of iconographical objects satisfying some criteria (qualitative and quantitative) specifying by the teacher. Beside textual descriptions of the task to be solved, such a model contains teacher’s recommendations and exemplary plans for finding task solution. The plan represents a hierarchical decomposition of the task to sub-tasks and possible solutions for each sub-task found the teacher herself. Each solution is represented by a sequence of SPARQL queries sent by the teacher to the library and some restrictions on the number of objects that should be selected as proper answers on the query (see [13] in the current issue for more details). The learning task model is created by the teacher by means of Learning Task Editor (a component of the SINUS tool layer) and used by the SINUS TEL application for monitoring and assisting the student in the process of solving the corresponding task. The repository is implemented as a MySQL relational database.

3.2. Semantic Annotation Editor

The Semantic Annotation Editor (SAE) is the SINUS tool intended for creating and editing semantic annotations of multimedia objects based on the specialized ontologies stored in the SINUS ontology repository. It can be used by the user for creating his/her own semantic annotations of objects stored in a digital library with a fixed system for accessing its content (that is fully determined by the library annotation schema). The SAE has been developed as a tool for operating with the SINUS extended SOA-based digital library. From this point of view its main user is a “librarian” who is responsible for updating and maintaining the library resources, as well as for providing access for end-users – “readers” to objects stored in the library.

[3] http://sinus.iinf.bas.bg/files/SINUSBasicOntology.owl
[4] http://195.96.244.8:8080/sinus-demo/ontology.jsp?ont=SIO
[5] http://sinus.iinf.bas.bg/files/SINUS_ImageSpecOntology.owl

In some non-educational applications of SAE, the librarian and the reader could be the same person. For example, according to the SINUS scenario for exploitation of the basic library [10], the possible users of SAE are specialists on different aspects of the East Christian iconography, who use the Editor (and the extended digital library as a whole) for creating and maintaining their own specialized collections of multimedia documents ("libraries") assisting their research.

Considering the SAE as a part of the Learning System (the developed TEL Demonstrator for collection development), in which two traditional user roles (the teacher and the student) exist, the Editor is a tool of the teacher, who uses it for preparing some learning resources to be used then by students.

The SAE is implemented as a complex RESTful Web service consisting of two main modules (also implemented as RESTful services) – Graphical SPARQL Designer and Semantic Annotator.

3.2.1. Graphical SPARQL Designer

The Graphical SPARQL Designer (GSD) is a Web service providing a graphical interface for communication between the heterogeneous repositories of SINUS and their users. The Designer, along with SINUS repositories and the Extended Search Engine (ESE), form the extended SOA-based digital library of SINUS . The users of the GSD can be not only humans, but programs as well – for example, it is invoked by the Learning Task Editor during creation of task models and also used by the Learning System as the main instrument of a student for solving the learning task formulated by the teacher.

The GSD is implemented as a set of modules realizing such functions as user identification and authorization, configuration of ontologies, creating search queries, creating visualization queries, forming object collections, and communications with external services.

As it has been already mentioned, in SINUS environment searching for multimedia objects is based on their annotations made by means of ontologies. However, in order an OWL ontology to become accessible for all SINUS users it should be configured. The configuring is organized as a process consisting of several steps, including:

- Determining possible concepts to be searched. This step allows the user to specify what ontology concepts (classes), which in the general case constitute a part of some hierarchical descriptions of other more complex concepts, can be directly accessible for creating search queries.
- Determining such properties of the possible ontological concepts to be searched that will be served as default ones for visualization purposes when a concrete instance of such concepts will be retrieved from the SINUS repositories. In SINUS environment when the user formulates a query he/she is able to specify not only properties that queried objects should have but also what properties of such objects she wants to see in visualization of query results. The default properties used for visualization significantly facilitate the end-user in their work with the SINUS extended library since they allow automatically displaying the important

(from the librarian point of view) properties of objects formulated by the user in the search query.

- Determining connections between the basic and the specialized ontologies. As it has been mentioned in the previous section, the user can search the SINUS repositories by means of queries formulating in terms belonging to several ontologies describing such objects from different points of view. Since the specialized ontologies are extensions of some concepts (partially) described in basic ontologies such correspondence between specialized and basic concepts should be preliminary specified. The connections are set by the librarian during the process of ontology configuration.

There is another aspect of possible relations between specialized and basic ontologies in SINUS environment. We assume that some values of specialized semantic annotations of concrete multimedia objects (so called "preliminary semantic annotations") could be automatically extracted from certain basic annotations of such objects represented as unstructured texts (see next section for details). That is why the ontology configuration process includes the step, in which the librarian could specify which properties of basic concepts used for creating basic annotations of multimedia objects could be explored for creating preliminary specialized annotations.

Although the process for searching objects with desired properties is implemented internally in SINUS environment via sending SPARQL queries, the construction of such queries remains hidden for the end-user. The user creates his/her query in a graphical way by following the structure of the selected ontologies and by choosing possible values for some terminal ontology elements (such as data-type properties or class individuals) provided by the Designer in a lexical natural-language form. As an addition the Designer also generates a natural-language like representation for each query constructed by the user that facilitates understanding and checking the meaning of the query.

Since we assume that the user of the Designer is a humanitarian that could have problems with formulating a complex query that includes too many restrictions on the desired properties of objects to be searched, it is possible to send a query to the results on another previously constructed query. The Designer provides the user with the history of his/her search represented as a list of natural-language represented queries the user sent along with sets of objects selected by the user as results for such queries. That is why the process of constructing a complex query is realized as a two-steps process: at the first step the user selects a set of objects from the search history that satisfies the restrictions described in the corresponding query and then constructs a simple query to that set specifying what additional properties a desired subset of this set of objects should have.

Another interesting and innovative feature of the Designer is an ability of the user to specify preliminary what properties of the objects satisfying the current query should be displayed in the result when such objects will be found. The aim is to facilitate the user in her choice of the desired objects from the list with query results. Before sending the query for searching objects with the desired properties the user creates a query for visualization selecting from the graphically visualized ontology structure the properties she wants to be displayed when the object will be found. The selected properties are added to the default properties for visualization preliminary chosen for such an object by the librarian during the process of configuring ontologies[6].

3.2.2. Semantic Annotator

The semantic annotation process in SINUS environment can be understood as a process for creating RDF-based annotations of concrete information objects belonging to some OWL classes described in the ontologies stored in the SINUS ontology repository. The main characteristic of this process is that although an ontology provides the user with a "pattern" describing all properties an object could have along with (in some cases) the range of all possible values for each property, the filling of such "patterns" should be carried out "manually" by the annotator-human. When the number of objects to be annotated is big, this procedure becomes very time-consuming. That is why we have tried to reduce the amount of such "manual" work by dividing the annotation process on two phases – the creation of the preliminary annotations executed automatically by means of SINUS environment, and the creation of the final annotations carried out by the annotator-human.

The preliminary annotation process is a process for automatically creating some semantic annotations for objects specified by the user that is based on available information about these objects represented as unstructured texts. The main part of this process is the analysis of the texts by means of some Natural Language Processing (NLP) techniques (see [11] in the current issue). It is important to note that although preliminary annotations look like semantic annotations "manually" created by the annotator-human, they have the following underlying characteristics:

- *Potential incompleteness* – the quality of the preliminary annotations to a great extent depends on the quality of texts used for their creation. Such texts could contain information only on some subset of ontological properties of objects to be annotated. Moreover, in some cases the texts themselves that are considered as a potential source of terms used for creating preliminary semantic annotations could be missing. Thus the preliminary created semantic annotations are incomplete in principle.
- *Potential incorrectness* – the preliminary annotations are created by application of some NLP techniques that, due to the high complexity of natural languages, are neither universal nor absolutely correct. That is why the preliminary annotations could contain imprecise or even erroneous information.

The described-above reasoning explains why the process for creating preliminary annotations should be followed by the next step – creating the final semantic annotations carried out by the human – expert in the concrete problem domain who is able to fill or correct incompleteness or incorrectness made in the preliminary annotations. The preliminary semantic annotation process is executed

[6] The user can also delete some (or all) of these default properties.

only once for each configured specialized ontology and applied to all multimedia objects described in SINUS basic libraries. The automatically created preliminary semantic annotations are temporary stored in SINUS repository for new semantic annotations [12].

The final semantic annotations are created by the user of the Semantic Annotator module (the “librarian”) via a graphical interface that is very similar to that used for constructing SPARQL queries. The librarian constructs a query describing objects she wants to annotate using terms from a selected specialized ontology, and the module retrieves such objects from SINUS repositories. The preliminary annotations are shown to the librarian as default values for object properties described in the specialized ontology and she can confirm or change these values. If such default values are missing for some object properties (i.e., no preliminary semantic annotations covering these properties have been created), the librarian can specify a desired value via selecting it from the list with possible property values presented in the corresponding ontology.

Finally the selected property values along with the corresponding properties are formed as INSERT/UPDATE SPARQL queries that are sent to Extended Search Engine component of SINUS environment for execution. The created final annotations are stored or replace the corresponding preliminary annotations in the Repository for new semantic annotations.

3.3. Learning Task Editor

The Learning Task Editor (LTE) is a component of SINUS environment intended for creating and editing learning tasks in some humanitarian problem domains and is implemented as a RESTful Web service. According to the SINUS exploitation scenario one of such tasks is the development of an analytical project analyzing some pre-assigned by the teacher aspects of the problem domain [10]. The general plan recommended by the teacher for solving this complex task proposes to decompose the task onto three simpler sub-tasks: 1) constructing a limited-sized dedicated collection from multimedia objects stored in the SINUS repositories with semantically annotated resources; 2) analyzing the prepared collection by comparison and debate of certain objects characteristics, and 3) combining the analysis and the collection into a multimedia document. At the moment the LTE allows the teacher to prepare all information needed for defining the first learning sub-task for selecting a multimedia object collection.

The task is described by its model containing both textual description of the task and a plan proposed by the teacher for solving it. In its turn the plan is represented as a sequence of steps, where each step contains textual description of actions to be done, qualitative constraints on the step results represented via SPARQL queries formulated by the teacher in his/her attempt to solve the step and some quantitative constraints on the required size of the step results. The detailed explanation of the model used for describing the collection development task, as well as the description of the Learning Task Editor structure and functions can be found in article [13] of the current issue.

3.4. Extended search engine

The Extended Search Engine (ESE) is a complex WSDL Web service used as middleware between the tool layer and the storage layer of SINUS environment. The main function of the ESE is to find answers on SPARQL queries sent by the Graphical SPARQL Designer. Since in SINUS environment the description of an object is represented in a distributed way (i.e., as semantic and non-semantic annotations stored in different heterogeneous repositories) finding objects satisfying a complex query is not a trivial task. In the general case such a process consists of the following phases:

- Analyzing the SPARQL query and converting it into a sequence of SPARQL sub-queries that will be sent to the corresponding SINUS repositories. Each sub-query asks only for such properties of objects mentioned in the initial query that are presented in the corresponding repository. Thus a set of objects satisfying the sub-query is a superset of objects satisfying the whole query.
- Executing the sub-query sequence – the answer of each sub-query is used as a filter for the next sub-query in the sequence. In other words, each next sub-query in the sequence is sent to a set of objects found by the previous sub-query.

In order a SPARQL query sent to a SINUS repository to be executed it should be translated into the "language" understandable by the corresponding repository. For example, since according to the SINUS exploitation scenario the basic SINUS repository is represented as a WSDL service, such query should be translated into a set of corresponding service functions which arguments are described according to the service metadata schema. This task of "lowering" semantic representation of a query to a non-semantic level understandable by the corresponding non-semantic repository is carried out by a dedicated service, which is a part of the ESE.

In some cases in order to find an answer on a SPARQL query sent to the SINUS semantic repository certain information available only in a non-semantic form is needed. In such cases a specially designed "lifting" procedure (service) which is another part of the ESE is used. Using the metadata schema and the basic ontology of the corresponding non-semantic repository as parameters, the procedure constructs RDF-based semantic representation of requested objects [12]. Such a representation then can be temporary inserted into the semantic repository to infer the answer to a SPARQL query sent to it or used for other purposes (e.g., for visualization of object properties).

The ESE is able not only to find an answer on SPARQL queries but also to create new objects described as SPARQL queries. Such functionality is used by the Semantic Annotation Editor for creating new semantic annotations.

A special module of the ESE is intended for extracting preliminary semantic annotations from texts written in a natural language (in our case – in Bulgarian). The module interface is implemented as a RESTful service allowing other services of SINUS environment to use the third-party CLaRK system [14]. CLaRK contains a set of instruments for structuring, processing and monitoring data represented as XML documents. In the SINUS context CLaRK is invoked by the Semantic Annotation Editor, which sends as parameters an OWL ontology, containing ontology terms with natural language labels, and a list of unstructured texts to be analyzed. The system returns the same set of texts extended by partially created semantic annotations represented in XML format, which then transformed to preliminary RDF-based annotations stored in the Repository for new semantic annotations (see article [11] in the current issue for details).

4. Evaluation of SINUS environment

This section evaluates the realization of SINUS environment and makes a comparison of the achieved results against the project goals.

4.1. Integration of heterogeneous data

The use of a relational database system for implementation of the "primer" library chosen to play a role of basic source for domain knowledge in SINUS environment as well as representation of other domain knowledge as OWL ontologies and RDF-based semantic annotations have posed to the developers a requirement for selecting an appropriate strategy for integration of all domain knowledge in an effective and consistent manner. Such integration should allow finding the desired objects and making logical inference needed for acquiring new knowledge in real time. In attempt to solve this problem the following possible approaches have been considered:

- RDF transformation – in this approach a relational database is transformed to RDF representation and then stored in the semantic repository along with other semantically represented sources [15]. Such transformation is done only once and can be implemented as a Web service. The main problem with such an approach is the replication of data and the synchronization of original database with its semantic copy. It is necessary either to implement a special procedure for actualizing RDF data on a certain interval of time or to develop a special notification mechanism that is activated when some changes in the original database have occurred. Such changes have been expected to occur comparatively frequent in the case of SINUS since the primary digital library would exist and still being extended outside the SINUS environment.
- Virtual RDF views – in this approach a relational database is not changed while its RDF-based view is published [16] An example of such approach is D2R Server[7]: the database content is mapped to RDF by a customizable mapping which specifies how resources are identified and which properties are used to describe resources. Based on this mapping, D2R Server allows a RDF representation of the database to be browsed and searched. The SPARQL interface enables applications to query the database using the SPARQL query language over the SPARQL protocol. The server takes requests from the Web and rewrites them via a D2RQ mapping into SQL queries against a relational database. This on-the-fly translation allows clients to access the content of large databases without having to replicate

[7] http://d2rq.org/d2r-server

them into RDF. However, such created virtual RDF views should be then integrated with other semantically presented data.

- RDF federation – this is an approach allowing a collection of heterogeneous distributed RDF repositories to be accessed as if they are a unique local semantic repository [17]. The conception of federating for executing distributed queries is not new [18] and based on an idea for the presence of an additional service – mediator that is responsible for distribution of sub-queries to the corresponding repositories and then for integrating the sub-queries results. It is assumed that such a mediator has required additional knowledge about all repositories.

The SINUS approach may be considered as a combination of the virtual RDF views and the RDF federation methods. The SINUS Extended Search Engine plays the role of a mediation service as it does in the RDF federation method. It decomposes an initial SPARQL query to a sequence of SPARQL sub-queries to be sent to the corresponding repositories. Each sub-query is constructed from the terms described in ontologies (basic or specialized) associated with the corresponding SINUS repositories. The special mapping procedures "lower" such semantic sub-queries to the Web service functions querying the corresponding basic digital libraries and then "lift" query answers back to a semantic level (RDF views). As in the RDF federation approach such a constructed semantic view is used either as a filter for the next query in the sequence or temporary inserted into the semantic repository to be used along with its content as an integrated source of semantic data to infer an answer to the query.

4.2. Correspondence to the project research goals

One of the project goals is to develop an environment allowing "upgrading", integrating and using existing heterogeneous digital multimedia libraries implemented as Web services. The developed SINUS environment completely satisfies this goal allowing the "upgrade" to be made into three directions.

- Extending domain knowledge – in SINUS environment new knowledge can be added by:

1. Constructing extended basic ontologies describing in an ontological manner all objects presented in the corresponding basic libraries. In such a way it is possible to add new knowledge that can be inferred from the basic domain knowledge (e.g., it allows to introduce a new concept "hard material", which is a generalization of such basic concepts as "stone", "wood", "glass", etc., mentioned in the basic library).
2. Adding new ontologies describing concepts and properties that have not been used explicitly in the object descriptions stored in basic libraries. In SINUS environment such new aspects of domain knowledge are represented as specialized ontologies.
3. Adding new knowledge about *concrete* multimedia objects stored in the basic libraries. In SINUS environments new knowledge of that type are represented as new semantic annotations describing the objects by specialized ontology terms.

It is important to note that in all three cases the extension of basic domain knowledge does not change any information stored in the basic digital libraries.

- Extending the way of searching for objects stored in the existing basic libraries by using new knowledge or their combination with basic domain knowledge. In SINUS environment the object search process is organized via intensive use of ontologies. A search query is constructed through tracing ontologies describing the properties of an object that has been selected to be searched while the SINUS approach for binding specialized and basic ontologies allows their using in combination. In such a way one and the same object can be presented to the user as having different properties depending on ontologies selected for formulating the search query. It allows the same query construction interface to be used by different groups of specialized users, for example, students specialised in theology or those interested in iconographic techniques or icon restoration, etc. The SINUS approach for splitting all knowledge to basic and specialized allows the users to build their own systems for indexing and searching for objects stored in external libraries without any changes in their content.
- Enhancing the way for displaying the search results by an ability to explicitly specify what additional information the user wants to be shown. The main advantage of using ontologies for searching information is that in contrast to other "traditional" approaches the SINUS user knows in advance the complete structure (i.e., all possible properties) of objects he/she wants to find. As a result the user can explicitly specify not only properties that the searched objects should obligatory have, but also such additional properties that would influence the user in her final decision what objects to select from the list of results returned by the search engine. This feature of SINUS environment aims at shortening the boring process of inspecting long lists with results that are usually returned by traditional search engines. The user can easily select potential candidates for the final solution just taking a look on such important from her point of view properties of each object from the list.

The second goal of SINUS project is to extend the developed environment by some tools facilitating the implementation of active learning approach in such humanitarian domains where the multimedia digital libraries are intensively used. In order to achieve the goal we have developed a set of tools devoted to two main user groups participating in the learning process – teachers and students.

The Semantic Annotation Editor (SAE) and the Learning Task Editor are the tools intended for using by the teachers. The SAE allows creating and editing semantic annotations of objects stored in a digital multimedia library. Such annotations are then used for searching objects described by a combination of their properties and represented as a search query. Although SAE can be used outside the learning context (as a tool of an annotator-"librarian" responsible for maintenance of a digital library) when SINUS environment is used for development of TEL applications this tool is used by the teacher. Our experience in developing such applications[8] has shown that in order to be usable by the end-users, who are not computer scientists, the tool should satisfy the following main requirements:

[8] See, for example FP6 Logos project – www.logosproject.com.

- Easy-to-use interface employing just problem domain terms. In the case of SINUS this requirement has been fully satisfied since the SAE is provided with an intuitive graphical interface, which is based on domain ontologies. The annotation process is implemented as a natural extension of the “traditional” for SINUS environment process for querying libraries that is realized via very similar interface.
- Flexible way for organizing the work allowing the user to determine by herself the order and the amount of work to be done. The SAE user can easy specify what objects he/she wants to annotate at the moment by sending a search query describing the necessary properties the objects should have. Moreover, in order to avoid the boring process for filling complete object annotation patterns the user can select what part of the corresponding pattern (defined by its ontological description) she is willing to fill and what part – to leave unfilled (or to fill later). In all cases the created (partial) semantic object descriptions remain syntactically correct and can be immediately used for searching the objects.
- Facilitating the user in carrying out his/her work by providing with all useful information available in the system. The annotation process (especially when many objects should be annotated) is associated with entering a big amount of information. This makes the process very boring and time-consuming for the user. In order to facilitate and speed-up this process the SAE exploits the approach for automatic creation of preliminary semantic annotations, which is based on extracting information from unstructured texts by means of some NLP methods [9]. Since these methods do not guarantee the absolute correctness of the results such created annotations are considered in the SAE as preliminary ones, i.e., they have to be confirmed by the user – human. Nevertheless, the ability to propose some preliminary semantic annotations has been evaluated by some teachers from the National Art Academy, who tested SINUS environment, as of great help to the SAE users.

Another SINUS tool – the Learning Task Editor (LTE) has been specially designed for learning tasks related to work with digital libraries. One of the important tasks for developing analytical skills of students-humanitarians is the task for selecting illustrative material (multimedia objects) from SINUS repositories with semantically annotated resources that serves as a basis for writing a course project analyzing different aspects of the selected multimedia objects. The LTE allows the teacher to formulate such learning tasks as the tool automatically extracts from the created task description all criteria needed in the future for checking the correctness of solutions of the same task proposed by students. Since the task for creating a collection of objects satisfying a set of requirement could have multiple correct solutions, in the LTE the teacher specifies only one of them. For that purpose he/she uses the dedicated means provided by SINUS environment (the Graphical SPARQL Designer) playing, in such a way, the role of a student attempting to solve the task. The tool analyzes the proposed solution and uses it for constructing the model for all possible corrects solutions of the task. Then this model is used for monitoring and assisting students when they are trying to solve the same task.

The application of the active learning principle not only to the process for solving a learning task by the students but to the process for creating the learning task description by the teacher as well, has been highly appreciated by the teachers participating in the LTE testing. The described-above approach has allowed replacing a rather difficult for formalization process of specifying all criteria that should be satisfied by a correct task solution, by a simple illustration of a possible correct solution. A side effect of this approach is a possibility of the tool to check the correctness of the solution proposed by the teacher herself making the comparisons between teacher's recommendations intended to the students and concrete steps the teacher has done in attempting to solve the task. Although the Learning Task Editor is just an experimental prototype system, which interface could be significantly improved, from the functional point of view the tool completely satisfies the research goals of the projects related to technology enhance learning.

A TEL Demonstrator for the learning task of collection development has been implemented as an illustration of learning abilities of SINUS environment. It is a learning system assisting the students to create multimedia collections satisfying a set of criteria formulated by the teacher. The system allows the student to solve learning tasks specified by the teacher in three different modes of operation:

- *Fully unaided* without any use of the exemplary plan for solving the task proposed by the teacher. In this mode the system checks for correctness and evaluates only the final solution of the task found by the student.
- *Independently but having in mind the teacher's plan for decomposition of the initial task to sub-tasks*. In this mode the system checks for correctness and evaluates only partial task solutions, i.e., the solutions of the sub-task found by the student. The way for solving a sub-task is not fixed and depends only on the creativity of the student and her abilities to work with the system.
- *By following the detailed plan for solving the task proposed by the teacher*. This is the case when the student has found it difficult to find the proper solution of the task and needs maximum assistance from the system. The assistance is the detailed plan describing not only how to decompose the task to sub-tasks, but how each sub-task could be solved. However, even in this case the student has the freedom to execute each proposed step of the plan in her own manner while the system checks for correctness and evaluates only the final results of each step.

In all three modes of operation the check for correctness is based on the task model constructed in the Learning Task Editor by analyzing the task solution proposed by the teacher. The student's solution is checked against the following criteria:

- *Correctness* – whether the set of objects proposed by the student as a solution of a concrete step (sub-task or task) contains some objects that do not satisfy (a part of) conditions (restrictions) formulated in a query (or queries) suggested by the teacher for solving the corresponding step.
- *Completeness* – whether a set of objects proposed by the student as a solution of a concrete step (sub-task or task) contains all possible in the moment objects satisfying conditions (restrictions) formulated in a task model constructed

from a query (or queries) suggested by the teacher for solving the corresponding step.

In such a way the types of student errors detected by the systems correspond to such classical in the Information Retrieval characteristics as precision and recall [19].

At the moment the check for possible causes of the detected incorrectness or incompleteness of a student solution is reduced to finding an erroneous elementary step in the plan applied by the student for solving the corresponding part of the task (see [13] for details). This restriction is due to the fact that the evaluation is based only on the mapping of the task model built from the plan for finding the solution proposed by the teacher against the set of objects proposed by the student as a solution. In the future in order to detect what part of the erroneous elementary step is incorrect we are going to analyze also the corresponding SPARQL query used by the student to carry out this step.

The developed TEL Demonstrator for collection development has proved the correctness of the developed approach for constructing TEL applications in SINUS environment and it has shown that the environment completely satisfies the project goals.

5. Conclusion

The developed SINUS environment contains three types of components – repositories used for storing different types of information; tools used this information for developing TEL applications, and a middleware carrying out the communications between the tools and the repositories. The environment has a high degree of heterogeneousness: at the moment it comprises three types of repositories – a relation database used for storing learning task models, a WSDL Web service implementing all functions of an SQL-based basic digital library storing multimedia objects and their annotations, and a semantic RDF-based repository storing OWL ontologies and semantic annotations of objects described in the basic digital library. The environment components are written in different programming languages such as C# (Extended Search Engine) and Java (Semantic Annotation Editor, Learning Task Editor) that are implemented as WSDL or RESTful Web services. The successful use of such heterogeneous tools for creating the exemplary TEL applications (the extended digital library with the service oriented architecture and the learning system in the domain of the East Christian iconography) has proved the correctness of the chosen approach for developing the SINUS environment.

It is necessary to remind that the SINUS project goals were neither to develop a concrete extended digital library for Christian iconography nor to build a concrete system for technology enhanced learning in this problem domain. That is why the graphical interfaces of the developed experimental prototypes are quite far from the “industrial” standards for communicating with real users of similar commercial systems. The same note may be addressed to the completeness of the developed ontologies describing the illustrative problem domain. However, those prototypes have proved the correctness of the chosen approaches for realizing the project goals and can serve as a basis for developing future really industrial systems for operating with heterogeneous digital libraries and for developing TEL-based applications.

All research software, as well as OWL ontologies developed within the frame of SINUS project are freely available for research purposes and can be found at the project site sinus.iinf.bas.bg.

Acknowledgments: The work on this article was partially funded by the Bulgarian National Science Fund project D-002-189 SINUS "Semantic Technologies for Web Services and Technology Enhanced Learning".

R e f e r e n c e s

1. D o c h e v, D., G. A g r e. Supporting Learning-by-Doing Situations by Semantic Technologies. – In: D. Stefanoui, J Culita, Eds. Proc. of 17th Annual Conf. on Media and Web Technology EUROMEDIA'2012, Bucharest, April 2012, ISBN 978-90-77381-69-4, 49-53.
2. Protege 4 User Documentation (accessed 2010). **http://protegewiki.stanford.edu/wiki/Protege4UserDocs**
3. P a n e v a-M a r i n n o v a, D., R. P a v l o v, M. G o y n o v, L. P a v l o v a-D r a g a n o v a, L. D r a g a n o v. Search and Administrative Services in Iconographical Digital Library. – In: Proc. of the International Conference "Information Research and Applications" (i.Tech 2010), July 2010, Varna, Bulgaria, 177-187.
4. OWLIM Version 5.3 Documentation. (accessed 2012). **http://owlim.ontotext.com/display/OWLIMv53/Home**
5. M a r i n c h e v, I. Lifting and Lowering the Data from Digital Library "Virtual Encyclopaedia of Bulgarian Iconography". – In: Proc. of 12th International Conference on Computer Systems and Technologies (CompSysTech'2011), Vienna, Austria, 16-17 June 2011, ACM, ISBN: 978-1-4503-0917-2, 179-184.
6. SPARQL 1.1 Query Language. W3C Proposed Recommendation 8 November 2012. **http://www.w3.org/TR/sparql11-query/**
7. E u z e n a t, J., P. S h v a i k o. Ontology Matching. Springer, 2007.
8. S t a y k o v a, K., I. H r i s t o v. Metadata Models for Technology Enhanced Learning in SINUS Project. – In: Proc. of the International Scientific Conference on Information Communication and Energy Systems and Technologies (ICEST'2010), Vol. **1**, 23-26 June 2010, Ohrid, Macedonia, ISBN 978-9989-786-57-0, 337-340.
9. S t a y k o v a, K., G. A g r e. Use of Ontology-to-Text Relation for Creating Semantic Annotation. – In: Proc. of 13th International Conference on Computer Systems and Technologies (CompSysTech'2012), ACM, New York, 2012, 64-71.
10. P a v l o v a-D r a g a n o v a, L., D. P a n e v a-M a r i n o v a. A Use Case Scenario for Technology-Enhanced Learning through Semantic Web Services. – International Journal "Information Technologies & Knowledge", Vol. **3**, 2009, No 3, 257-268.
11. S t a y k o v a, K., P. O s e n o v a, K. S i m o v. New Applications of "Ontology-to-Text Relation" Strategy for Bulgarian. – Cybernetics and Information Technologies, Vol. **12**, 2012, No 4, 43-51.
12. M a r i n c h e v, I. Semantic Lifting of Unstructured Data Based on NLP Inference of Annotations. – In: Proc. of 13th International Conference on Computer Systems and Technologies (CompSysTech'2012), ACM, New York, 2012, 58-63.
13. A g r e, G., D. D o c h e v, L. S l a v k o v a. A SINUS Approach to Technology Enhanced Learning for Humanities by Active Learning. – Cybernetics and Information Technologies, Vol. **12**, 2012, No 4, 25-42.
14. S i m o v, K., Z. P e e v, M. K o u y l e k o v, A. S i m o v, M. D i m i t r o v, A. K i r y a k o v. CLaRK – An XML Based System for Corpora Development. – In: Proc. of the Corpus Linguistics 2001 Conference, 2001, 558-560.

15. Use Cases and Requirements for Mapping Relational Databases to RDF. W3C Working Draft 8, June 2010.
http://www.w3.org/TR/rdb2rdf-ucr/#directPlusOnt
16. B i z e r, C., R. C y g a n i a k. D2RQ Lessons Learned. Position Paper for the W3C Workshop on RDF Access to Relational Databases. Cambridge, USA, 25-26 October 2007.
17. J a é n, J., A. B o r o n a t, J. H. C a n ó s. Federated TDF Repositories for Integrated Hybrid Museums. – In: Proc. of International Conference on Digital Culture and Heritage (ICHIM'2005), Paris, 21-23 September, 2005.
http://www.archimuse.com/publishing/ichim05/Jaen.pdf.
18. S h e t h, A., L. L a r s o n. Federated Database Systems for Managing Distributed, Heterogeneous and Autonomous Databases. – ACM Computing Surveys, Vol. **22**, 1990, No 3, 183-236.
19. B a e z o-Y a t e s, R., R.-N. B e r t h i e r. Modern Information Retrieval. Pearson Education Limited, 1999, ISBN 0-201-39829-X.