Abstract: This paper proposes an approach for direct transformation of a part of ontology of the national development fund in the user interface components. Generating and designing user interfaces are reduced to a formalized semantic description of administrative tasks. The main advantages of the proposed concepts are: a) the integration of elements and the unification of representations of user interface components, b) easier maintenance and more efficient generation of user interfaces; c) the possibility of efficient generation of alternative, functionally equivalent user interfaces.

The proposed approach, with standardized formalism in the method of creating ontology of the administrative process (task), can be applied to any administrative process, thereby significantly reducing the time of modeling, testing and integrating user interfaces. Following the proposed approach, the semantics of an administrative task of the Guarantee Fund of Autonomous Province of Vojvodina (APV) is described in details.

Key-Words: eGovernment Ontology, User interface, Services

1 Introduction

Knowledge of business processes, documents and participants in business processes and decision-making within the given domain is a collection of expertise, knowledge based on experience and individual knowledge and skills of administrative staff that are used in the conduct of administrative tasks.

The model should meet the following conditions:

1. To provide an explicit representation of knowledge of administrative processes (tasks).
2. To explicitly present the elements that make it possible to provide the groundwork for creating a user interface (UI) as a component of the information system on the basis of knowledge of business processes and documents.

The main advantages of the model to generate a user interface which will be described hereinafter is a representation of knowledge about the domain which, in the use of traditional techniques for developing information systems, are firmly integrated into the user interface code by a separate component of formal ontology and a standardized user interface.

The paper is organized as follows. The second section provides an overview of the related work. The third section presents an ontological model of the administrative task. The fourth section shows the practical use of ontologies designed to generate the user interface. In the last, fifth section, concluding remarks and directions for further research are given.

2 Related work

This section will present research results related to the problem investigated in this paper. Modern trends in the development of information systems, specifically in the development of user interfaces, are based on the use of tools that will release developers from low-level programming. The main objective of generating a user interface based on an ontological model is an automatic translation of high level models that describe the user interface into the executive code of applications.

Generally, the user interface has the following tasks: to present information to the user and to support user-system interaction. For building
ontologies (user interface supported ontologies), authors in [4] propose the following criteria:

Determination of the domain described by the ontology (real world or IT system)

1. The complexity of the ontology (informational, low, medium or high complexity)
2. The temporal aspect of use of the ontology (design phase or phases of exploitation)
3. The manner of presenting the ontology to the user (graphical representation, no presentation...)
4. The way of the interaction between user and ontology

Based on the analysis of the criteria, the authors define three main reasons for the use of ontologies in generating user interfaces: 1. Improved visualization of UI; 2. Improvement of interactions between the system and users, 3. Improvement of the development process of UI. In [5], four basic principles are suggested to generate a user interface, which is based on an ontological model: 1) Each model component of the user interface should be represented by an ontological model. 2) Ontological models of interface components that are available over the Internet should be used to create models of the user interface whenever possible. 3) The user interface and the application should be designed and implemented as two different components. 4) Tools for user interface development should ensure the implementation of basic functions.

Each user interface has a set of user functions for its own purposes. These functions act as data entry, query resolving, reviewing of the results and application exit. Analyzing the benefits of ontologies creating formal descriptions of user interfaces, in [6] the authors propose the use of ontologies that will enable the integration of user interface and create the basis for the generation of modular integrated user interfaces.

Ontologies have been successfully integrated in several works. For example, in [7] the ontology supported the monitoring and requirements of development of the user interface, and in [8] ontologies are used as input models for an automatic code generator to create user interfaces. In a study of applying ontologies in the development of eGovernment [9] the authors presented a method of modeling ontologies in the domain of eGovernment. According to the authors [9] each public service of eGovernment is semantically modeled and contains references that point to the required input data. Predefined values of input data or preconditions can be expressed with semantic rules. In this way, it allows automatic creation of (web) forms and interactive validation of input data. According to the authors, the development of new applications and projects begins with modeling the ontology. The modeled ontology is a description of the domain that is used as the basis for creating new applications or services. In this context, the created ontology is comparable to the platform independent model (PIM) and as such is used in MDA (Model Driven Architecture).

The presented rules used for modeling ontologies, according to the authors [9], allow the unambiguous identification of e-Government services. The authors [10], proposed the use of algorithm for the direct transformation from OWL to relational data structures. The authors [11] also dealt with these problems. The basic idea is to transform the created ontology with the help of the transformation tools in the DLL script and thus preserve all relations, constraints, and information about the domain. DLL script is used to generate a relational database.

This paper will present the methodology of administrative task ontology construction and annotation and define the algorithm for direct transformation of the semantic description of the administrative task in the user interface components.

3 Annotating the ontology with elements of user interface

Figure 3.1. shows the detailed ontological model of the Administrative task of processing applications modeled in the Protégé tool [16], presented with RDF-Gravity, RDF Graph Visualization Tool [17].
Figure 3.3 presents an ontological model of the administrative task of processing an application. A semantic representation of an administrative task can be described as follows: The procedure `Application_processing` belongs to a class of Procedures. The Administrative Office executes the procedure `Application_processing`. The document `application_document_to_Open_Competition` is the result of the execution of `Application_processing` procedure. The procedure for application processing has a unique document template `TxtApplication`. The electronic service that calls the procedure `Application_processing` is described by the service profile `Pservice_A_P`.

The annotation of the ontology with elements of user interface is based on the following basic principles (rules): Administrative tasks have input data. As shown in Figure 3.1, input data are a series of variables. The names of the input data are represented with `DataProperties` of the observed task. Descriptions of input data used to generate the user interface are defined in `DataProperties - isDefinedBy` property in the following form:

```xml
<Control>Component_type</Control> ....
Type of the UI component (ComboBox, etc.)
<Data>data_type</Data> .....................
Data Type (string), (integer)
<Order>X</Order>
...............................Component index
order on the form
```

1. Individuals of each class denote the template of the document that is filled with input data during the execution of the administrative task and generate the final document. The feature `Data_Properties_assertion` references a document that represents the template (Doctxt).
2. `Data_Properties ID` defines the order of execution of the administrative procedures (tasks).

4 Automatic creation of user interface

The requirement set in the introductory section of this paper is that the created ontology should enable automatic generation of user interface. The basis of this procedure is the transformation of the created Administrative task ontology represented in RDF / XML format in two XML documents.

Administrative tasks sorted in the order of execution within the business process of issuing the guarantee described in 3.1 will are presented by a Tree View component of UI. For each administrative task, it is necessary to load the descriptions of user interface components from the property `DataProperty - isDefinedBy`. The Figure 4.1 graphically shows the model for automatic UI generation.
The first XML document OntoClass.xml is a representation of classes and subclasses of the ontological concept Procedure that represents an ontological description of the administrative business processes. OntoClass.xml is a XML document, which represents the Data Source for XML - tree control that is automatically created. The second XML document OntoForm.xml represents user interface components that are extracted from the semantic description of the administrative task. The user interface will be created using the generated XML documents and associated XSL styles.

4.1. Semantic content transformation module

The module for transformation of semantic content shown in Figure 4.1 represents the application that loads the created ontology and executes SPARQL queries. Parsing the query results generates two XML documents (OntoClass, OntoForm).

4.2. Creating OntoClass.xml documents

The generation of OntoClass documents has two steps. In the first step the SPARQL query that has the task to generate a list of operative procedures (tasks) that are described in the ontology, whose sequence on the list is identical to the order of performance (ontological class Procedures, describe the workflow of the administrative procedures in the process of Guarantee issuance) is activated. The list of administrative task obtained by following code is placed in a temporary file tmpClass.txt. In the second step, a result file is parsed and the result of parsing is an OntoClass.xml document. OntoClass XML document is cited as data source of TreeView control. Figure 4.2. shows OntoClass.xml and a TreeView component of the user interface.

Listing 4.1. used SPARQL query:
Listing 4.1.
" SELECT * WHERE {"+
"?s rdfs:SubClassOfgf:Procedure }"+
"order by Classid }";

4.3. Creating OntoForm.xml documents

For generating OntoForm documents, it is necessary to perform the SPARQL query for each selected administrative procedure defined in the ontology and extract DataProperties of the selected operative procedures and properties DataProperties – isDefinedBy. The obtained results are stored in the tmpForm.txt file. A result file is
parsed and the result of the parsing is an obtained OntoForm.xml file. Previous text describes how to extract data for generating user interface from a modeled ontology. So far, the extraction of work procedures and the extraction of DataProperties and isDefinedBy property for the selected administrative procedure have been presented.

Listing 4.2. used SPARQL query:
Listing 4.2.
" SELECT ?property ?label WHERE {"+
"?property rdfs:domain@g:Application_processing."+
"OPTIONAL { ?property rdfs:isDefinedBy@l ?label }");"

4.4. Generating user interface

Figure 4.1 shows the process of generating a user interface. XSL document Form.xsl provides mechanisms of transformation and formatting OntoForm.xml and OntoClass.xml documents. Figure 4.4 shows the generated user interface. Screenshot of the generated user interface is shown below.

Figure 4.3 Generated user interface

5 Conclusion

Creating and using ontologies of knowledge of administrative processes and modeling systems that would speed up and automate the work of state administration, creates prerequisites for technical and organizational interoperability of different government agencies. In addition, an extremely important aspect is the flexible and cost-effective mechanism for creating a user interface that provides interaction of various types of users (employees in the administration, employees of business entities and citizens) with the e-Government system. The proposal of the model for ontologically based generation of the user interface as described in this paper represents the idea to standardize the representation of user interface. Further research should focus on the development of ontologies of the administrative processes within the domain of state bodies. In this way, we would ensure, among other things, the basis for fast and efficient creation of an ontological model for the user interface of information systems of state administration standardized in the aspects of interaction, visualization, maintenance and generation.

References:


