Katarzyna DĄBROWSKA-KUBIK[1]

Warsaw University of Technology(1)

doi:10.15199/48.2015.03.04

DevOntoCreator: an ontology-based approach to software engineering

Abstract. *This article goal is to present an innovative method to improve the implementation phase of a software development process. This is achieved by an automatic code generation using an ontology definition and an explicit and implicit knowledge of software engineering and web application constructing. The author of this paper proposes the Intelligent Software Engineering using Semantic Network (ISE-SemNet) with the DevOntoCreator tool, which facilitates software engineering. Thanks to ISE-SemNet semantic web, it is possible to increase a software engineering knowledge, which is accessible and understandable, either to human programmers or machines.*

Streszczenie. *Celem tego artykułu jest przedstawienie innowacyjnej metody na ulepszenie fazy implementacyjnej procesu tworzenia oprogramowania. Metoda ta jest oparta na automatycznym generowaniu kodu źródłowego tworzonej aplkacji z wykorzystaniem ontologii, jak również jawnej i ukrytej wiedzy z zakresu inżynierii orogramowania oraz budowania aplikacji webowych. Autorka artykułu proponuje wykorzystanie sisci semantycznej ISE-SemNet (Intelligent Software Engineering using Semantic Network) wraz z narzędziem DevOntoCreator. Dzięki nim jest możliwe wzbogacenie wiedzy na temat fazy implementacyjnej procesu wytwarzania oprogramowania, która jest łatwo dostępna i zrozumiała zarówno dla ludzi jak i maszyn* ***(Narzędzie DevOntoCreator: zastosowanie ontologii w inżynierii oprogramowania)***

Keywords: DevOntoCreator, ontology, semantic web, software engineering, ISE-SemNet
Słowa kluczowe: DevOntoCreator, ontologia, sieć semantyczna, inżynieria oprogramowania, ISE-SemNet

Introduction

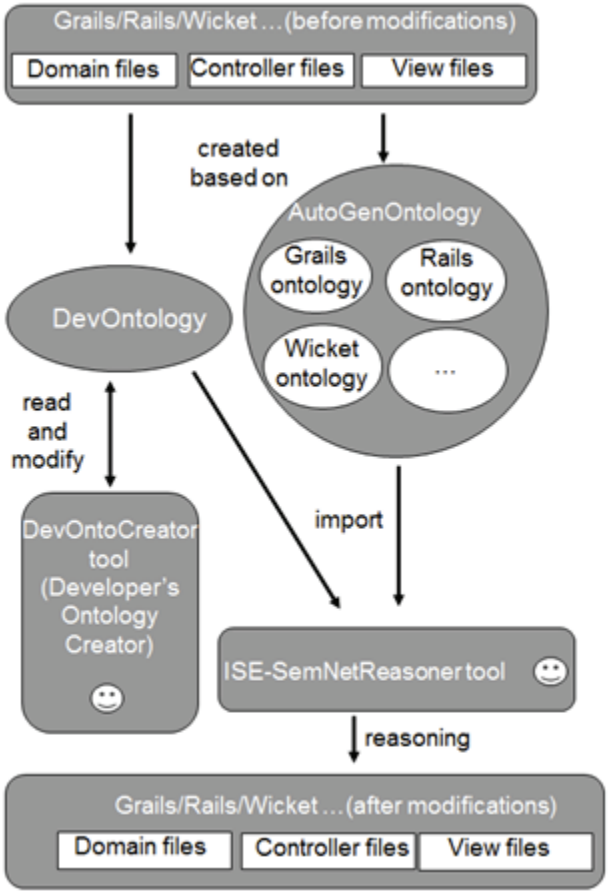

Fig. 1. The architecture of the ISE-SemNet semantic web.

Nowadays, there are a lot of frameworks, which relieve programmers of arduous and laborious works during software development process, for example GRAILS (Groovy on Rails) [6], RAILS (Ruby on Rails) [7], Wicket [8].

Unfortunately, the way these frameworks work arouses controversy. They give satisfying results, when supporting the simple, repeatable activities, but supporting the difficult, complex operations make troubles. There are two cases, when the generated code is not satisfactory. The first one is changing by hand the source code generated by the tool. The second one is implementation without machine's help, like using MVC frameworks. There are several tools described in the literature, such as KontoR [1], which present a number of different approaches for using ontologies in the context of Software Engineering in the different stages of a software engineering life-cycle [5].

The author of this paper proposes the DevOntoCreator tool, which is the most important part of semantic web, called ISE-SemNet (Intelligent Software Engineering using Semantic Network) [9]. ISE-SemNet is a semantic network, which represent semantic relations between tools, such as the DevOntoCreator and the ISE-SemNetReasoner, ontologies, such as the AutoGenOntology and the DevOntology as well as a web application, which the programmer wants to develop. In this way, the knowledge about software engineering process, and activities performing by programmers and machines, will be gathered in ontologies of this system.

Complex and differentiated software applications, created with the DevOntoCreator tool, are going to be better, faultless and more reliable. Moreover, time of software development is going to be shorter. The other advantages of this solution are common understanding of the development process, comprehensive information representation and communication between people in the software development organization [9]. This paper is structured as follows: the author first describes the GRAILS framework. In section 3 the architecture of the ISE-SemNet semantic web is derived. Details of the ISE-SemNet ontology is described in section 4. In section 5 the author of this article presents the DevOntoCreator tool. Finally, the section 6 describes conclusions.

What is GRAILS?

GRAILS (Groovy on Rails) is an open source Model-View-Controller framework for building web applications, which is based on the Java language [6]. It uses Groovy as a programming language and is compatible with the Convention Over Configuration [6]. This paradigm strives for decreasing the source code even on very complex software. GRAILS hide a lot of application configuration from programmers. In this way, the need of configuration and coding is reduced to minimum. Moreover, the time of software development is much shortened – what is possible, when programmer observes the GRAILS rules.

It is worthwhile to point, that GRAILS makes use of templates to create domain, controller and view source codes. The default template can be modified by the programmer,

as far as it is necessary. The following command "grails install-templates" copies the templates used by Grails during code generation to the project directory. Then, the folder /src/templates/scaffolding is created in the project directory, which contains artifacts templates and application skeletons. Next time on the grounds of them, GRAILS will generate a skeleton and artifact templates. First of all, GRAILS checks, if there are templates modified by the user. In this way, the application is generated using the modifications. Otherwise, the default template for GRAILS is used. The GRAILS framework generates applications with the following CRUD operations: create, read, update, delete. This functionality is called scaffolding, which decreases the source code of a created application. Additionally, Grails generates a database schema, GSP views and the controller behavior for the given domain classes. Thanks to them, the programmer can give an attention to crucial artifacts and activities [6].

Architecture of the ISE-SemNet system

The architecture of the ISE-SemNet semantic web and placement of the DevOntoCreator tool is shown below (Fig. 1). In this semantic web, there are two ontologies, which are used as the foundation to the environment: the AutoGenOntology and DevOntology. "An ontology is a logical theory accounting for the intended meaning of a formal vocabulary, in other words, ontological commitment to a particular conceptualization of the world. An ontology consists of concepts and relations, and their definitions, properties and constrains expressed as axioms" [2] [3] [4]. These ontologies don't intend to describe all the knowledge about Software Engineering, but only an area of Model-View-Controller (MVC) frameworks, such as GRAILS (Groovy on Rails) [6], RAILS (Ruby on Rails) [7], Wicket [8], and so on.

The first one is the AutoGenOntology, which contains a set of skeletons for various software environments, based on MVC pattern, such as the Grails Ontology, Rails Ontology or Wicket Ontology. It describes all concepts, activities and the source code automatically generated by the MVC framework. This part of ontology should be initiated before the ISE-SemNet enviroment started. On the other hand, the DevOntology involves all software developers' activities, including the source code and modifications on concepts from the AutoGenOntology [9].

The DevOntoCreator is a tool dedicated for software engineers to modify the source code on the ontology level, without changing of created application's files. The most important part of the system is the ISE-SemNetReasoner. It generates output files without a programmer's attention based on following methods:

- input files of the created application,
- ontologies,
- the SCM Algorithm (Source Code's Modification), invented by the author of this article.

The SCM Algorithm describes how to modified output files taking into consideration programmer's modifications saved in the DevOntology and MVC frameworks' skeletons, saved in AutoGenOntology. The ISE-SemNet reasoner is executed in the last phase of the software development process using the ISE-SemNet system. The last part of the ISE-SemNet semantic web is source files of the developed application, appropriate before and after modifications, dependent on a chosen framework, such as GRAILS. Only the ISE-SemNetReasoner tool can modify input files and present them as output files to the programmer, as it was written before.

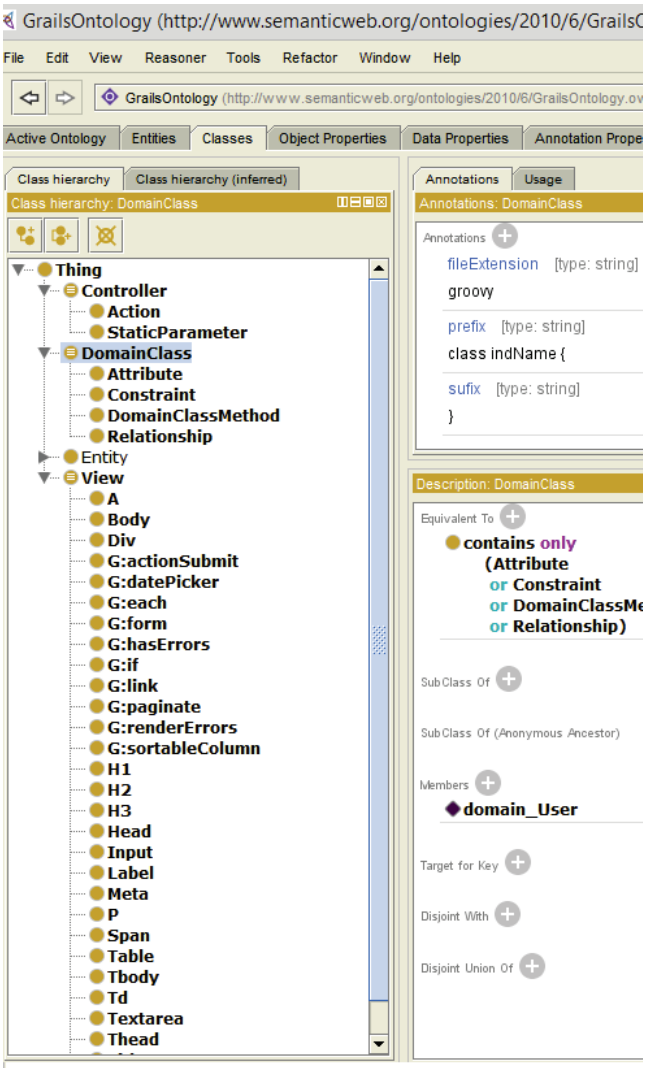

Fig. 2. Class hierarchy of the DevOntology and AutoGenOntology in the Protégé 2000.

AutoGenOntology and DevOntology

The class hierarchy of the ISE-SemNet ontology was created based on types of MVC applications' files: *DomainClass*, *Controller* and *View* (Fig. 2). The domain class represents a model of a problem and can generate the database schema automatically. The controller is used to send commands to the model and the view. It modifies the model's state and changes view's presentation. The view requests information from the model, that it uses to generate an output representation to the user. The *Attribute*, *Constraint*, *DomainClassMethod* are subclases of the DomainClass. The Controller has two subclasses: *StaticParameter* and *Action*. Finally, HTML tags, such as *Head*, *Body*, *Span*, *A*, *Div* and Groovy tags, such as *G:form*, *G:if*, *G:each*, *G:paginate*, fulfill the subclasses role of the View class (Fig. 2).

The DevOntoCreator tool during the developed application files proccessing makes use of three type of properties of the ISE-SemNet ontologies, which are described below. The first group of properties is *Data Properties* (Fig. 3). In this group there are AutoGenOntology's properties, which describe parameters of HTML and Groovy tags. They are:

- *groovyProperty*, such as *var*, *total*, *property*, *precision*, *controller*, *bean*,
- *htmlProperty*, such as *charset*, *cols*, *content*, *for*, *href*, *style*,
- *htmlGroovyProperty*, such as *action*, *id*, *name*, *class*,

onClick, *value*, *url*, *title*, *method*, *status*.

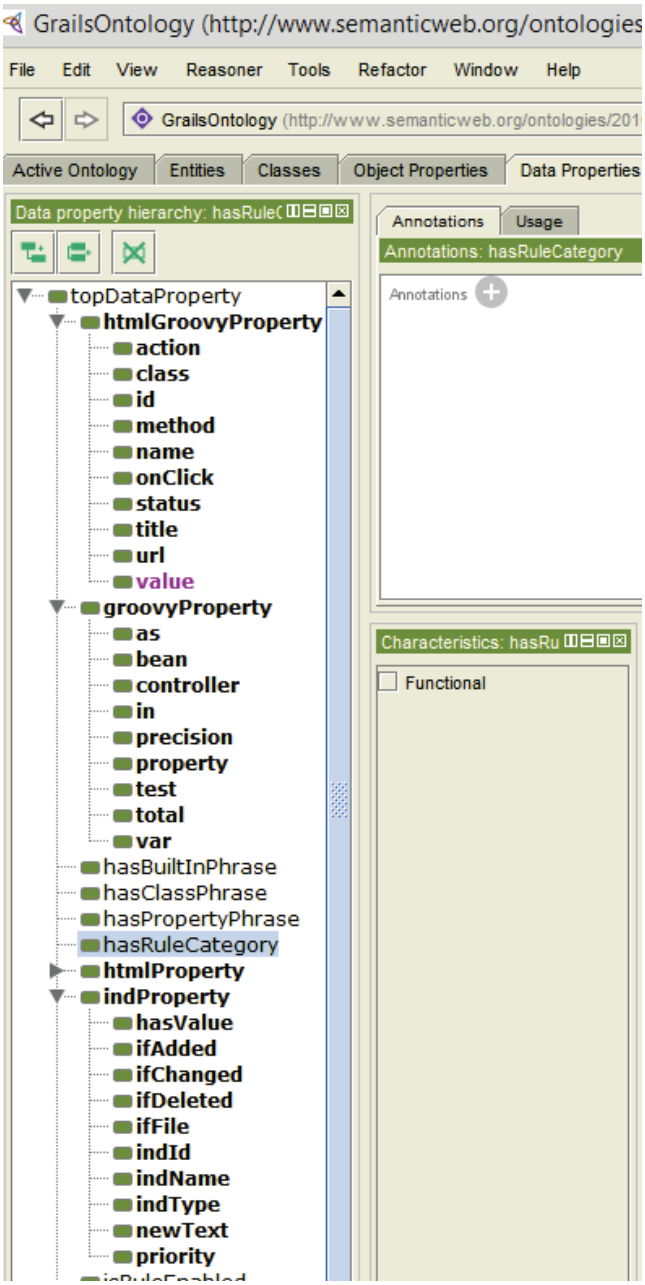

Fig. 3. Data property hierarchy of the DevOntology and AutoGenOntology in the Protégé 2000.

The second group of the *Data Properties* is *indProperty* (Fig. 3). In this group there are DevOntology's properties, which are used to identified the fragment of the code of the developed application, like *indId*, *indName*, *indType* and *priority*. The *newText* and the *hasValue* properties keep a new value and an actual value of the fragment of the developed application's file, respectively. Finally, *ifAdded*, *ifChanged* and *ifDeleted* properties are used to diagnose whether the given block of the file was just added, changed or deleted by the programmer. The *priority* property decides about the order of the each block of the developed application's source code. Finally, the *ifFile* property assumes *true* value in the event that DevOntoCreator tool modifies the file's parameters, not its content. View's files created in MVC frameworks have nested layouts tags. Therefore the author of this paper creates the pair of the inverse *Object Property*, which is a part of the DevOntology: *contains()* and *inContainedIn()*. Their details is presented below in the OWL2 language, where *GO* is abbreviation from *GrailsOntology* :

```
<!– http://www.semanticweb.org/ontologies/GO.owl#
contains –>
<owl:ObjectProperty rdf:about="&GO;contains">
<rdfs:domain rdf:resource="&GO;DomainClass"/>
<owl:inverseOf rdf:resource="&GO;isContainedIn"/>
<rdfs:subPropertyOf rdf:resource="&GO;
tagObjectProperty"/>
</owl:ObjectProperty>

<!–http://www.semanticweb.org/ontologies/GO.owl#
isContainedIn –>
<owl:ObjectProperty rdf:about="&GO;isContainedIn">
<rdf:type rdf:resource="&GO;FunctionalProperty"/>
<rdfs:range rdf:resource="&GO;DomainClass"/>
<rdfs:subPropertyOf rdf:resource="&GO;
tagObjectProperty"/>
</owl:ObjectProperty>
```

The last group of properties is *Annotation Property*, which includes:

- *prefix* – defines starting string, includes identifier, such as "class User {",
- *suffix* – defines ending string, such as "}",
- *fileExtension* – defines file extension string, such as "gsp" or "groovy".

These properties are example of the AutoGenOntology. They are set only once for classes and will not be changed during software development process by the programmer.

The DevOntoCreator Tool

The DevOntoCreator supports programmers in creating and the maintenance of software using MVC frameworks [6] [7] [8]. Its goal is not to substitute development environments (DE), like Eclipse [10] or Visual Studio [11], but improve them. In this paper the author suggests using it as a plug-in DE. It is worthwhile to emphasize, that the date processing in the DevOntoCreator tool is based on ontologies, not on configuration files.

The scenario of use involves the following steps:

1. Choose the framework (i.e. Grails, Rails). Now, the AutoGenOntology, which describes this framework, is loaded automatically (Fig. 4).
2. Choose the and appropriate DevOntology.
3. Choose the directory of the application, that you want to develop.
4. Choose one of the program mode, such as Domain, Controller or View, depending on which files user wants to modify.
5. Now the program shows all files of the given type (DomainClass, Controller or Views) on the "File list" in a tree view (see on the left side of Fig. 4). Additionally, the tool offers three options: "Show file", "New file" and "Delete file". Thanks to them, the user can displays or edits chosen file, and adds new one.
6. If the user chooses the "Show file" option, the content of a file is shown in the middle of the DevOntoCreator. Furthermore, it offers the following options: "List individuals", "Show individual", "New individual", "Edit individual" and "Delete individual" (Fig. 4). An individual is the block of the chosen file, which corresponds to an instance of the concept in the DevOntology. The Option "List individuals" divides the chosen file on individuals,

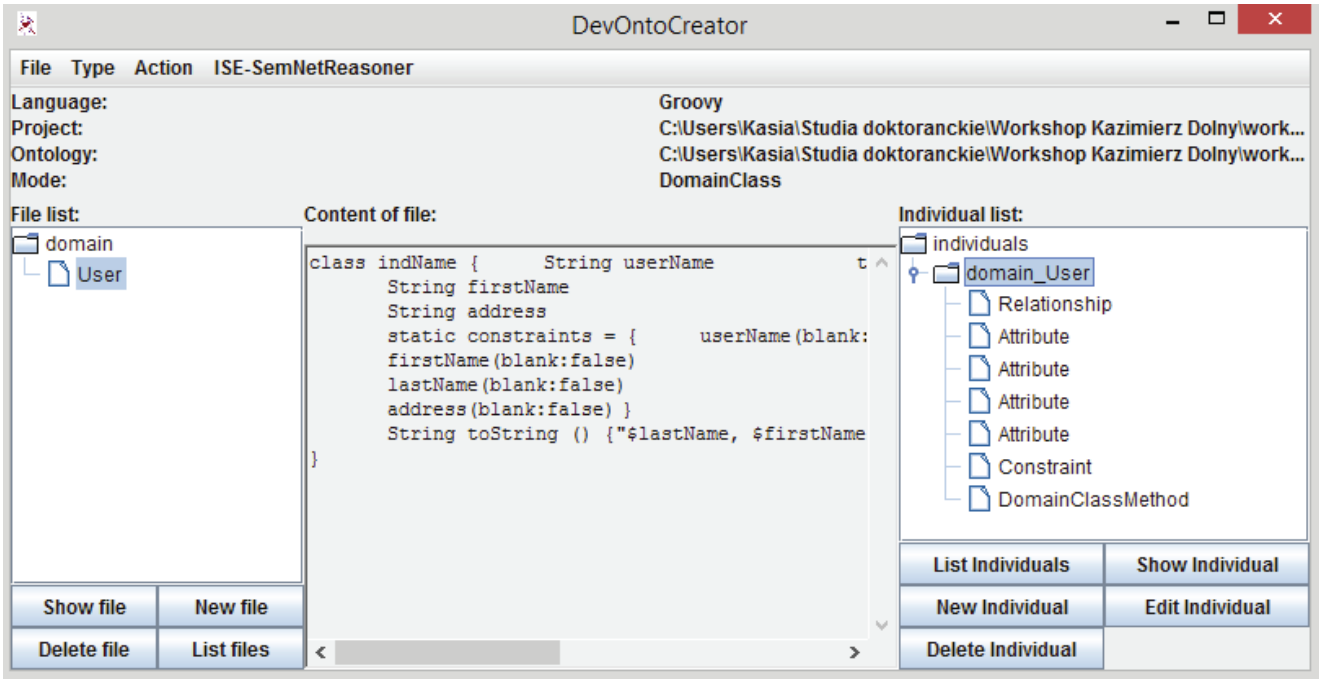

Fig. 4. DevOntoCreator tool – mode: DomainClass, framework: Grails.

which are display on the "Individual list" in a tree view (Fig. 4). These individuals are instances of a DevOntology that belong to the chosen file. Thanks to the "Show individual" option, it is possible to display the block of the file throughout choosing the instance from the "List individual" tree. Additionally, there are also "Edit individual", "Delete individual" and "New individual" options, which edit, delete or add the block of the source code. The example of the window, which appears after choosing the "New individual" option is shown below (Fig. 5)

Implementation

The architecture presented in the previous section has been implemented using a free and an open source Java framework called Jena [12]. The Apache Jena is an API for building semantic web and Linked Data applications [13] based on processing RDF's graphs. The data exchange between the DevOntoCreator tool and the DevOntology takes place throughout the Jena. As it has been mentioned before, the DevOntology is changed by the DevOntoCreator tool in the background.

The ontology of the ISE-SemNet system has been implemented using Protégé 2000 [9] [14] and using Pellet as the DL Reasoner [16]. In order to give the possibility of the DevOntology's changing by the programmer using the DevOntoCreator tool, the following methods were implemented:

- *boolean saveInd(IseInd t)* – for "Edit individual" option,
- *boolean deleteIndvdual(String indUrl)* – for "Delete individual" option,
- *boolean addInd(String mode, String dir, String fileName, String choosenAttribute, IseInd newInd, IseInd fileInd, IseInd parentNode)* – for "New individual" option.

It is worthwhile to point out that one of the most challenging method for the implementation of the DevOntoCreator tool, is *IseInd getInd(String indUrl)*. This method reads all necessary data about the given individual to the memory of DevOntoCreator program as the *IseInd* object. Then, these objects are used to the following option of implementation: "List individual" and "Show individual". Additionally, the DevOntoCreator process data from the DevOntology and the AutoGenOntology using complex algorithms, invented by the author of this article.

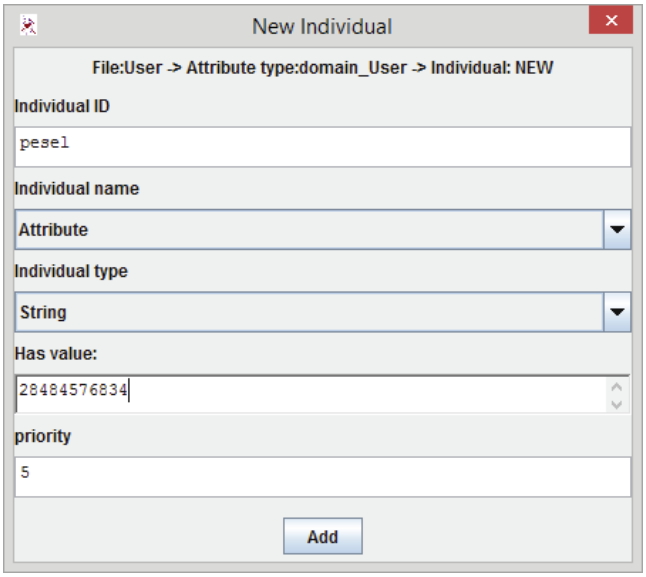

Fig. 5. The DevOntoCreator tool – "New individual" window

Summary and conclusion

The ontology-based tool, called DevOntoCreator, can be used to improve the software development process. Its main goal is to add new knowledge to the ISE-SemNet ontology about software engineering, at runtime and development time, which is accessible, both to human developers and machines. The author of this paper proposes an innovative method, which support software development process by automatic code generation using ontology as a knowledge base for the ISE-SemNet semantic web. Consequently, the quality of the source code increases, because its big part is created automatically. The time of development process is shorter, because the programmers don't have to repeat the same work. Despite of modeling ontologies is a tedious and costly task, the author of this paper believes, that this approach can be helpful for software developers working on

MVC frameworks. Finally, the ISE-SemNet is an open source and is scalable, so adding a new framework, such as Struts, to the ISE-SemNet isn't troublesome [15].

For future work the author of this paper plans to enrich the ISE-SemNetReasoner tool by adding the infer mechanism for improvement of its work. In other words, a set of rules in the SWRL language will be created, which takes into consideration the hierarchical layout of View files [17]. Nowadays, only the SCM Algorithm is implemented. The author predicts, that thanks to this solution looking for fragments of a given file, which were added, modified or deleted, will be much easier.

Moreover, the author will execute two experiments. The first one uses different MVC frameworks, such as GRAILS, RAILS, Wicket and Struts for building the same simple application in each case. The second one will be executed only using GRAILS framework, but different type of applications will be created. In the result, the author of this article will be able to give conditions, when software development process using ISE-SemNet system will be really helpful.

REFERENCES

[1] Happel H-J., Korthaus A., Seedorf S., Tomczyk P.: KOntoR: An Ontology-enabled Aproach to Software Reuse, in: proc. of The18th Int. Conf. On Software Engineering and Knowledge Engineering, 2006.

[2] Falbo R.A., Natali A.C.C., Mian P.G., Bertollo G., Ruy F.B.: ODE: ontology-based software development environment, IX Congreso Argentino de Ciencias de la Computación, Red de Universidades con Carreras en Informática (RedUNCI), pp. 1124–1135, Oct. 2003.

[3] Guarino N.: Formal Ontology and Information Systems, in Formal Ontologies in Information Systems, Proceedings of the first international conference (FOIS'98), June 6-8, Trento, Italy, IOS Press, pp. 3-15, 1998.

[4] Falbo R.A., Menezes C.S., Rocha A.R.C.: A systematic approach for building ontologies, in Proc. of the IBERAMIA'98, Lisbon, Portugal, 1998.

[5] Happel H-J., and Seedorf S.: Applications of Ontologies in Software Engineering, International Workshop on Semantic Web Enabled Software Engineering SWESE'06, Athens, USA, Nov. 2006

[6] Judd Ch.M., Nusairat J.F., Shingler J.: Beginning Groovy and Grails, ISBN: 978-1-4302-1045-0, Apress, 2008.

[7] C. Carneiro Jr., R. Al Barazi: Beginning Rails 3, ISBN: 978-1-4302-2433-4, Apress, 2010.

[8] Gurumurthy K.: Pro Wicket, ISBN: 978-1-59059-722-4, Apress, Sept. 2006.

[9] Dąbrowska-Kubik K.: Inteligentna automatyzacja wytwarzania oprogramowania przy użyciu sieci semantycznej ISE-SemNet, Zeszyty Naukowe Wydziału Elektroniki, Telekomunikacji i Informatyki Politechniki Gdańskiej, Nr9, Seria ICT Young, pp. 217–222, 2011.

[10] The Eclipse Foundation, 2014. [web page] https://www.eclipse.org,

[11] Visual Studio - MSDN Microsoft, 2014. [web page] http://msdn.microsoft.com/vstudio,

[12] I. Dickinson: Meet Jena, a Semantic Web Platform for Java, [web page] http://www.devx.com/semantic/Article/34968/1954, 2007,

[13] Getting started with Apache Jena, [web page] http://jena.apache.org, The Apache Software Foundation, 2011–2014.

[14] Protégé - a free, open-source ontology editor and framework for building intelligent systems, [web page] http://protege.stanford.edu, Stanford University Center for Biomedical Informatics Research, 2014.

[15] Welcome to the Apache Struts project, [web page] http://struts.apache.org/, The Apache Software Foundation, 2000-2014.

[16] Pellet DL Reasoner, [web page] http://pellet.owldl.com/,

[17] SWRL: A Semantic Web Rule Language Combining OWL and RuleML, [web page] http://www.w3.org/Submission/SWRL/,

***Authors**: MSc. Katarzyna Dąbrowska-Kubik, The Institute of Computer Science, Faculty of Electronics and Information Technology, Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warszawa, Poland, email: k.dabrowska@ii.pw.edu.pl*