Semantic Web Journal submission, tracking #: 1192-2404 (2016) 1–17
IOS Press

SeMFIS: A Flexible Engineering Platform for Semantic Annotations of Conceptual Models

Editor(s): Philippe Cudré-Mauroux, Université de Fribourg, Switzerland
Solicited review(s): Guido Governatori, NICTA, Australia; Three anonymous reviewers

Hans-Georg Fill *
University of Vienna - Research Group Knowledge Engineering, Waehringerstrasse 29, 1090 Vienna, Austria
E-Mail: hans-georg.fill@univie.ac.at

Abstract. In this paper, we present SeMFIS – a flexible engineering platform for semantic annotations of conceptual models. Conceptual models have been used in the past for many purposes in the context of information systems' engineering. These purposes include for example the elicitation of requirements, the simulation of the behavior of future information systems, the generation of code or the interaction with information systems through models at runtime. Semantic annotations of conceptual models constitute a recently established approach for dynamically extending the semantic representation and semantic analysis scope of conceptual modeling languages. Thereby, elements in conceptual models are linked to concepts in ontologies via annotations. Thus, additional knowledge aspects can be represented without modifications of the modeling languages. These aspects can then be analyzed using queries, specifically designed algorithms or external tools and services. At its core, SeMFIS provides a set of meta models for visually representing ontologies and semantic annotations as models. In addition, the tool contains an analysis component, a web service interface, and an import/export component to query and exchange model information. SeMFIS has been implemented using the freely available ADOxx meta modeling platform. It can thus be directly added to the large variety of other modeling methods based on this platform or used as an additional service for other tools. We present the main features of SeMFIS and briefly discuss use cases where it has been applied. SeMFIS is freely available via the OMiLAB.org website at http://www.omilab.org/web/semfis.

Keywords: Conceptual Models, Semantic Annotation, Ontologies, Analysis

1. Introduction

Conceptual models have a long tradition in supporting the engineering and analysis of information systems [38,53]. They have been used for example in requirements engineering for enabling the communication between different stakeholders [42], in the context of model-driven software engineering for designing systems and generating code [11], for simulating the behavior of information systems [34], or as runtime models for supporting "dynamic state monitoring and control of systems during execution" [1][p.23].

To create conceptual models it is necessary to use a modeling language, which defines the syntax, semantics, and notation of the constructs to be included in the models in a formal way [6,32]. As of today, a large number of modeling languages are available - well-known examples include BPMN [45], UML [44], IDEF [47], EPC [50], or iStar [54]. Although some of these languages are quite extensive and cover a variety of aspects, it is often required in practice to adapt their semantic scope to individual needs. This may for example result from changes in legal regulations – e.g.

*Corresponding author, e-mail: hans-georg.fill@univie.ac.at

due to new reporting obligations that require the documentation of additional information [22] – or, because of new business requirements – e.g. when conducting benchmarks that require particular information structures for enabling industry-wide comparisons [13].

Although the semantic scope of a modeling language could be adapted through a re-design and re-implementation of the language, this is often unfavorable. Especially in industry scenarios where sometimes thousands of models have been created over long time periods and with considerable effort [46], changes in the underlying modeling language may lead to unexpected side effects. Even more so in the case of runtime models, where execution systems interact with the models controlling their behavior.

As a solution to this problem, *semantic annotations* have been introduced for conceptual models [12]. By linking elements of conceptual models to elements in ontologies, additional semantic information can be added without having to modify the underlying modeling language. Thereby, ontologies are chosen because of their nature as an "explicit specification of a conceptualization" [31] that can be shared [43] and that permits to process the annotations by machines [49].

In this paper we present SeMFIS (Semantic-based Modeling Framework for Information Systems) – a meta-model based engineering platform for semantic annotations of conceptual models that supports the representation and analysis of annotations with ontologies, and that can be adapted to arbitrary modeling languages and integrated with other tools and services. While previous publications have mainly focused on general requirements [15,12], on particular scenarios for using SeMFIS [13,14,16], and technical extensions [28,17], this paper discusses the modeling, query, and exchange functionalities provided by the most recent version of SeMFIS in detail.

The remainder of the paper is structured as follows. In section 2 we will give an overview of related tools for the semantic annotation of conceptual models. Subsequently, we describe the features of SeMFIS, including the provided modeling editors, query and import/export functionalities, and its extensibility in section 3. Section 4 outlines the architecture of SeMFIS based on the ADOxx meta modeling platform. In section 5 we describe two use cases for applying SeMFIS and discuss limitations of the tool in section 6. The paper is concluded with an outlook on the next steps.

2. Related Tools

Whereas the semantic annotation of textual, video, and audio resources has been discussed in a multitude of publications and has led to the development of several tools – see e.g. [36] for a recent evaluation – approaches and tools for annotating conceptual models are rare and often limited to specific types of conceptual modeling languages. Semantic annotators for textual or data resources such as Apache Stanbol[1], OntoText[2] and the like are not primarily directed towards the use for conceptual models. However, they may be used for specific parts of conceptual models, e.g. to determine concepts from an ontology in the labels of model elements that are expressed in natural language.

Regarding approaches that particularly address the semantic annotation of conceptual models, several tools can be found in the area of semantic business process management. The original motivation for semantic business process management was to combine business process management with semantic web services [33]. For this purpose software tools have been developed that permit to enrich business process models with semantic annotations and subsequently discover web services that match these semantic descriptions. Examples include the Maestro tool [7], an extension of the ARIS modeling toolkit [48], WSMO Studio [9], or Pro-SEAT [41]. However, as these tools are directed towards business processes, they are mostly limited to particular business process modeling and ontology languages. For example, Maestro and WSMO Studio support BPMN and the ARIS extension supports EPC. Maestro, ARIS and WSMO Studio use ontologies expressed in WSML/WSMO format, whereas PRO-SEAT supports OWL ontologies. PRO-SEAT works on models generated by the Metis platform, which permits to define arbitrary modeling languages. However, it is not integrated with the platform but rather an external tool for conducting semantic annotations based on Metis file exports. In addition, not all these tools are freely available thus hampering their use and further development in academic settings. From the mentioned tools only WSMO Studio is currently available under an open source license.

Although also the field of software engineering has discussed semantic annotations of conceptual models, e.g. for UML class diagrams using the XMI format [55] or for SoaML and ODM [37], correspond-

[1] https://stanbol.apache.org/
[2] http://www.ontotext.com

ing tools are to the best of our knowledge not publicly available.

Another direction that can be found in the context of conceptual models with semantic enrichments are tools that represent conceptual models as ontologies. In general, such approaches require the a-priori definition of mappings of a conceptual modeling language to an ontology language. Examples for tools following this direction include ICOM for creating and maintaining ontologies using conceptual models [29] and SemPeT, which provides an ontology for the semantic analysis of Petri nets [10].

In summary, the tools that have been investigated above for conducting semantic annotations of conceptual models are limited to very specific target domains. Furthermore, none of these tools with the exception of PRO-SEAT permits to create semantic annotations for arbitrary types of conceptual modeling languages. However, we could not find an openly accessible version of PRO-SEAT. In addition, none of the tools we found seems to offer interfaces for the programmatic access of the stored information using web interfaces. As far as could be assessed from the corresponding publications, none of the described tools is directed towards supporting industry scenarios with several thousands of models. A core requirement for such scenarios would be for example to provide highly performant storage solutions for the models, ontologies, and annotations.

3. SeMFIS Features

SeMFIS[3] is a flexible platform for engineering semantic annotations of conceptual models. The intention behind SeMFIS is to provide a link in the form of a software platform between the field of conceptual modeling and the field of ontologies. Thereby, benefits can be gained from both sides: on the one hand, ontologies provide formal information structures and reasoning mechanisms that can be used to enrich conceptual models. On the other hand, the semi-formal style and visual editors traditionally used for conceptual models facilitate the interaction, in particular for non-technical users. In contrast to other semantic annotation tools for conceptual models, SeMFIS does not require a specific type of modeling language or the modification of an existing modeling language. Due to the decoupling of the semantic annotations in separate annotation models, SeMFIS can be directly added to existing modeling methods without affecting their structure nor behavior. As SeMFIS is implemented based on the industry-driven, freely available ADOxx meta modeling platform [24], it also provides a highly performant persistence layer in the form of a relational database as well as interfaces and a scripting language for programmatic access. In section 3.1 we will describe the model editors provided by SeMFIS. These are used to visually describe semantic annotations and for visualizing ontologies. Section 3.2 gives an overview on the scripting and analysis functionalities in SeMFIS including the programmatic access using a web service interface. In section 3.3 we will present import and export interfaces, in particular the interface to the Protégé platform.

3.1. Model Editors

The semantic annotation approach used in SeMFIS is based on theoretical elaborations discussed in [12]. In order to decouple the annotation information from the underlying conceptual models and ontologies, SeMFIS adds a distinct information structure for describing annotations. In this information structure references to elements in conceptual models and to ontologies are established. These references are then linked using annotator elements, which define the type of annotation. In this way, neither the modeling language for conceptual models nor the specification of ontologies has to be changed.

For representing semantic annotations and ontologies, SeMFIS uses a meta model-based approach. As shown in Figure 1 the SeMFIS meta model comprises four model types: the semantic annotation model type, the frames ontology model type, the OWL ontology model type, and the term model type.

The semantic annotation model type provides 'model reference' and 'connector reference' elements that can be used to refer to elements in conceptual models. Similarly, it contains the 'ontology reference' element to establish links to ontologies. Via the 'annotator' element connections between the reference elements to conceptual models and the ones to ontologies can be established. In addition, the type of annotation can be specified via the 'annotation type' attribute. In the preconfigured version of SeMFIS the following annotation types are available: 'Is equal to', 'Is broader than', 'Is narrower than', 'Is instance of', 'Is subclass of', 'Is

[3] http://www.omilab.org/web/semfis

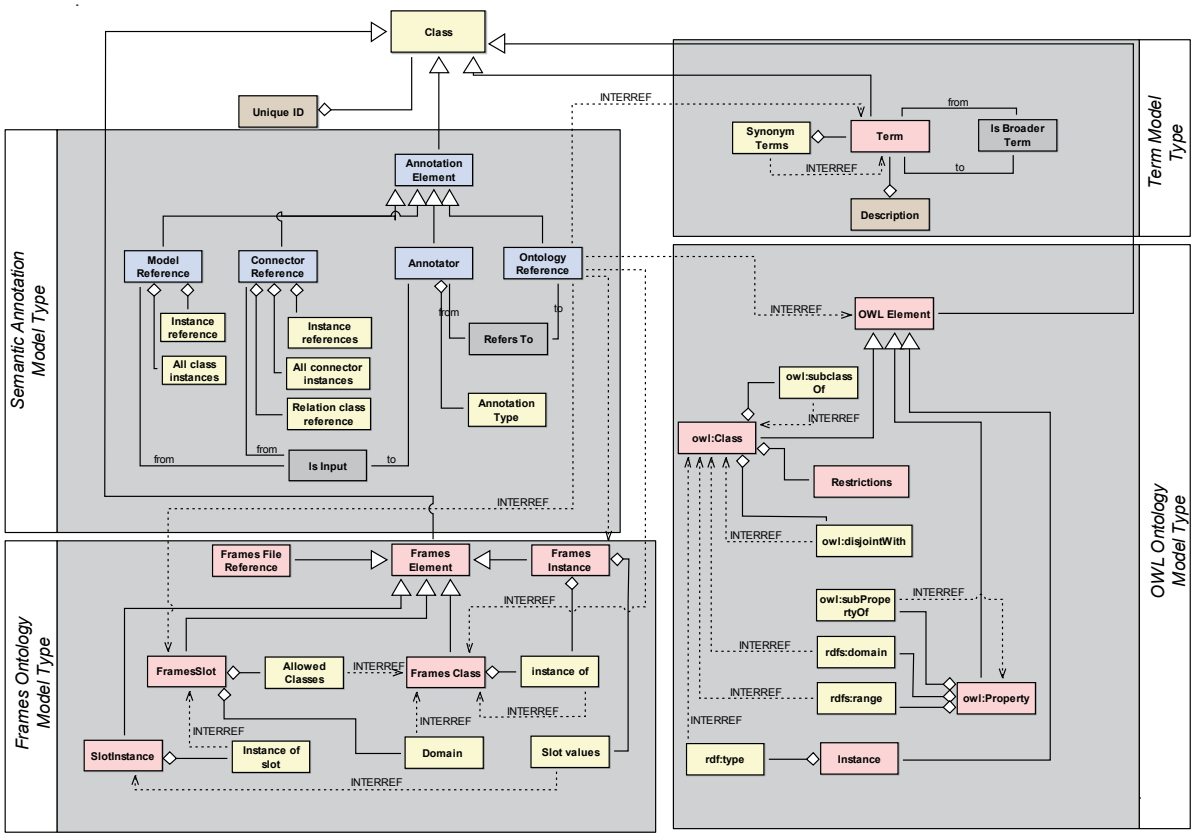

Fig. 1. SeMFIS Meta Model showing the SeMFIS Model Types

superclass of', 'Is instance using fromClass', 'Is instance using toClass'. By using so-called attribute profiles these types can be easily adapted or extended to fit individual needs.

In addition to the manual definition of semantic annotations by using the graphical representations, the scripting functionalities provided by SeMFIS – as discussed in the subsequent section – permit to automate such tasks. Such support functions however strongly depend on the target domain and the modeling language that is to be annotated and thus need to be individually designed - i.e. for suggesting annotations for supporting certain simulation tasks, as e.g. in [14], other kinds of recommendation algorithms are necessary than for annotations used in natural language processing, as e.g. in [35].

On the side of ontologies three different types of ontologies can be represented in SeMFIS: OWL ontologies that are today widely used based on the international standard issued by W3C, Frames ontologies as used in Protégé which follow the OKBC protocol [30], and controlled vocabularies in the form of the term model type which permits to express synonym and hypernym relationships between terms. For these ontology types SeMFIS provides the underlying information structures as shown in the meta model.

SeMFIS does however not provide reasoning mechanisms, rule engines, or further constraints regarding the processing of the semantics in the ontologies. If needed, these functionalities can be easily added to SeMFIS using third-party tools and APIs via the subsequently discussed interfaces and scripting functionalities. For using ontologies in the mentioned formats, they can either be manually defined in SeMFIS using the graphical model editors, imported via a plugin from Protégé as discussed in section 3.3.2, or using a mixed-mode of importing and manual adaptation.

For all the above described model types visual model editors are provided by the SeMFIS modeling component ( ). In this way the information contained in the SeMFIS model instances is graphically represented and users can interact with the information on a

visual level. In the screenshot shown in Figure 2 examples for model editors are shown. By double-clicking on the graphical elements, so-called 'notebooks' can be opened for editing the attributes of the model elements.

To establish the linkage between SeMFIS and other modeling methods, it can be chosen from two development directions. Either, the SeMFIS meta models are added to an existing ADOxx-based modeling method[4]. This has for example been done for the pre-configured version of SeMFIS that is publicly available on the SeMFIS website. As shown in the screenshot in Figure 2, SeMFIS has been added to simplified versions of the ADONIS business process management modeling method [24][p.19], BPMN diagrams [3], and UML class diagrams [4]. This also encompasses to inherit functionality from other methods, e.g. the simulation algorithms for business processes in BPMN that can be accessed in SeMFIS via the simulation component ( ).

For supporting the merging of SeMFIS with other modeling methods the ADOxx development environment can be used[5]. This environment offers graphical editors for specifying ADOxx meta models as well as linking different model types to each other. In the tutorial section of the SeMFIS website two tutorials for accomplishing such tasks are available [19,20]. The first tutorial shows how a new model type including classes and relationclasses can be added to the SeMFIS library [19]. The subsequent tutorial then illustrates how classes of the new model type can be added to the definitions of the model reference class, thus enabling their use in semantic annotation definitions [20].

For further easing the composition of different modeling languages, a Modeling Method Domain-Specific Language (MM-DSL) has recently been specified [51]. The purpose of this domain-specific language is to specify the requirements of modeling methods and their direct translation into a fully functional modeling tool [51][p.1]. Code generators for ADOxx which transform statements in this DSL to modeling method specifications that are executable on ADOxx-based tools are currently being implemented in a PhD thesis. First results of these developments have already been published [52]. The availability of MM-DSL together with its code generators will allow to develop further assistance mechanisms for composing modeling methods such as automated merging facilities or consistency checks of modeling methods.

As a second direction, SeMFIS can be used as a repository for storing and analyzing information about semantic annotations and ontologies without creating model editors. Then, linkages to elements in conceptual models have to be established using the interfaces and exchange mechanisms below.

3.2. Scripting and Analysis Functionalities

As SeMFIS has been implemented on top of the ADOxx meta modeling platform [24], it can re-use a number of functionalities that are provided by the platform's components. This includes scripting and analysis functionalities that have been made available for SeMFIS. Via the scripting functionality, statements in the domain-specific ADOscript language can be executed. This language permits to access almost any platform functionality in a programmatic way. It can thus be used for tasks such as constraint checking, model manipulation and analysis, report generation, interaction with third-party tools and DLLs, and user interaction via a set of pre-defined UI components.

ADOscript is a procedural, interpreted language. For executing scripts in this language, they can be assigned to items on the user interface such as buttons or menu items. In addition, scripts can be called via the notebooks showing attributes of model elements, graphical representations of model elements, or events that are triggered due to state changes of the platform, e.g. through user interaction or system events. For testing scripts at run-time SeMFIS offers a debug console, which is accessible in the modeling component under the menu item 'Debug'.

The central operations of ADOscript are available via so-called *message ports*. These receive messages as strings, execute the desired command and some of them return values. In total, ADOscript provides fourteen message ports. For SeMFIS the following eleven message ports have been made available: *Messageport AdoScript* which provides different types of interaction through dialogues on the user interface and the fundamental web service functionality, *Messageport Core* for executing basic operations on models such as the creation, deletion, and modification of objects, relations, attributes, and models, *Messageport CoreUI* which provides complex UI dialogues for se-

[4] See the websites `http://www.adoxx.org` and `http://www.omilab.org/web/guest/projects` that offer currently around 40 ADOxx-based modeling methods including methods for UML, BPMN, ER, Petri Nets, iStar and many others.

[5] For details see the ADOxx online documentation [5]

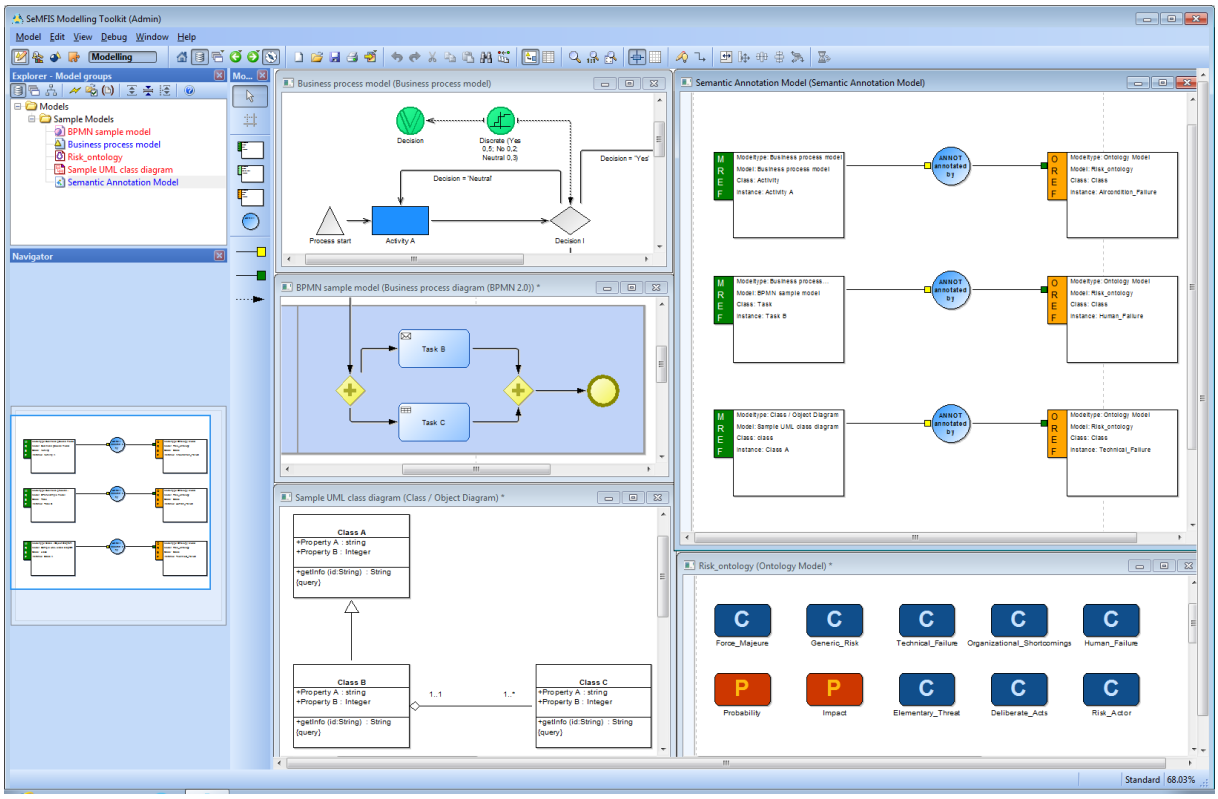

Fig. 2. Screenshot of SeMFIS showing Model Editors for a Business Process Models in ADONIS and BPMN notation, a UML Class Diagram, a Semantic Annotation Model, and an OWL Ontology Model

lecting models and attribute profiles, *Messageport Application* for querying and changing the behavior of the application, e.g. by adding menu entries and buttons, retrieving the current screen resolution, or enabling and disabling application components, *Messageport Modeling* for accessing and processing information contained in models, e.g. for generating graphics from models, printing models, or copying and moving objects, *Messageport Simulation* for programmatically executing simulations on models by using a set of pre-defined simulation algorithms, *Messageport ImportExport* for importing and exporting models, *Messageport Documentation* which provides support functions for creating documentations for models, e.g. for reading and writing XML files, *Messageport AQL* for executing queries in the ADOxx Query Language (AQL), *Messageport UserMgt* for creating users, changing user settings, etc., and *Messageport DB* for certain operations directly working on the used relational database, e.g. getting attribute values directly from the database.

In addition to the operations of the messageports, ADOscript code may also contain statements in the ADOxx-proprietary LEO language. LEO defines additional constructs such as an expression language, control structures, i.e. If-Else conditions, For/While loops, arithmetic operations, as well as data structures for interacting with arrays and maps. Furthermore, ADOscript defines constructs for specifying functions and procedures, for issuing calls to the operating system, and for dynamically loading and executing other scripts at run-time.

To illustrate the usage of ADOscript in SeMFIS we present in the following three examples. The first example shows the essential steps for creating a semantic annotation in a programmatic way. Thereby it is illustrated how ADOscript can be used to interact with model information. The second example shows how web services can be invoked via ADOscript to access external processing mechanisms. This is an essential feature to connect SeMFIS to other platforms and third-party tools. Finally, the third example will show how the SeMFIS SOAP interface can be made accessible. This is necessary if using SeMFIS in a web environment for remotely exchanging information with the platform.

For programmatically creating annotations, a user needs to specify a semantic annotation model to which the annotation should be added. ADOscript provides a set of pre-defined UI components that can be configured for individual purposes. When executing the code shown in listing 1, a corresponding list component is shown on the user interface. By specifying the model type for the model select box, the choice of possible models is restricted. After the user has chosen a model and left the dialogue, the identifier of the model is stored in a variable for further processing.

CC *"CoreUI"* **MODEL_SELECT_BOX**
boxtext: *"Please select a semantic annotation model"*
modeltype: *"Semantic Annotation Model"*

Program Code 1. ADOscript Statement for Creating a User Interface Component for Choosing a Semantic Annotation Model

In the next step, the selected model needs to be loaded and three elements have to be created: a *model reference* object, an *annotator* object, and an *ontology reference* object. With the model reference object, arbitrary objects in conceptual models are referenced. The ontology reference object links to arbitrary elements in OWL ontologies, Frames ontologies, or controlled vocabularies. And the annotator object stores the information about the type of annotation used. We show this exemplarily by the code for creating an object of the class Annotator in listing 2 with the CREATE_OBJ statement of the *Messageport Core*. It is assumed that the identifier of the model from the previous dialogue is stored in *selectmodelid* and the variable *classid* will receive the value of the id of the Annotator class.

CC *"Core"* **LOAD_MODEL**
  modelid: *(selectmodelid)*
CC *"Core"* **GET_CLASS_ID**
  classname: *"Annotator"*
CC *"Core"* **CREATE_OBJ**
  modelid: *(modelid)* **classid:** *(classid)*

Program Code 2. ADOscript Definition for Creating an Object Instance of the Class Annotator

Subsequently, we create connectors between the objects, e.g. as shown in listing 3 for connecting a model reference element to an annotator element. It is assumed that the identifiers of the model reference element and the annotator element are stored in the variables *modelreferenceobjid* and *annotatorobjdid*. The actual creation is accomplished with the CREATE_CONNECTOR statement of the *Messageport Core*. This statement takes as input parameters the originating object of a connector – specified in *fromobjid* – and the target object of the connector – specified in *toobjid*.

CC *"Core"* **GET_CLASS_ID**
  classname: *"is input for"*
CC *"Core"* **CREATE_CONNECTOR**
  modelid: *(modelid)*
  fromobjid: *(modelreferenceobjid)*
  toobjid: *(annotatorobjdid)*
  classid: *(classid)*

Program Code 3. ADOscript Definition for Creating a Relation of type *is input for* between a Model Reference Element and an Annotator Element

Finally, we show an operation on attributes by assigning a reference to an element. In the case of semantic annotations this is essential for actually establishing the link between elements in conceptual models and ontologies. It is done by accessing attributes of type Interref which contain references to other model elements similar to a pointer. In listing 4 this is exemplarily shown for creating the reference to a Class element in a UML class diagram. At first, the relevant attribute identfier is retrieved. This information is then used in the ADD_INTERREF statement for inserting the reference to the UML class named 'My UML Class'.

CC *"Core"* **GET_ATTR_ID**
  classid: *(modelreferenceclassid)*
  attrname: *"Instance reference"*
CC *"Core"* **ADD_INTERREF**
  objid: *(modelreferenceobjid)*
  attrid: *(attrid)*
  tmodelname: *"My UML Class Diagram"*
  tmodeltype: *"Class / Object Diagram"*
  tclassname: *"Class"*
  tobjname: *"My UML Class"*

Program Code 4. ADOscript Definition for Adding an Interref Attribute Value in a Model Reference Element

Another feature that is very important in the domain of semantic annotations for conceptual models is to connect to external services and platforms. In this way functionality from third parties can be re-used for pro-

cessing the information contained in conceptual models, their annotations, and ontologies. Furthermore, information may be handed over to other platforms and tools in the form of configuration or execution specifications, e.g. for reasoning, configuring ERP systems or for feeding information to workflow engines.

The example we show here is highly simplified in terms of the shown functionality for reasons of brevity - however, it conveys the central mechanisms of external service calls in SeMFIS. In essence, calls to third party tools typically require an intermediary component that invokes the operations at a remote endpoint.

We show this in the following for the case of SOAP-based web services where we require a component that is able to transfer SOAP request messages to a remote SOAP endpoint somewhere in the web. For the example we assume that we have annotated conceptual models with concepts from an ontology, which describes currencies. A typical use case would then be to convert cost information in an enterprise model to a different currency based on the current exchange rate. This shall be accomplished by using the public SOAP endpoint of webservicex.net that offers such conversion functions[6].

At first we have to retrieve the currency ISO codes from the ontology – see the code excerpts in listing 5. In the ontology used for the example, currencies are described in a way that is similar to the encodings used in DBPedia[7], i.e. using instances of an OWL *Currency* class that has an assigned datatype for the ISO code. The retrieval of the ISO codes is accomplished through the custom-developed ADOscript function GET_DATATYPE_PROPERTY_VALUE. Next, we have to compose a SOAP request message, which can be done using ADOscript variables. The variables *currencyfrom* and *currencyto* are thereby inserted in the SOAP request message and contain strings with ISO currency codes such as "USD", "EUR", "CHF", or "JPY" as defined in the ontology instances. The SOAP request is then stored in a temporary file.

This example could be extended by retrieving even more information from models or ontologies or by accessing more complex services. Examples for more complex interactions would the issuing of calls to a reasoning service while including the complete information of an ontology and corresponding instances in the SOAP request, the execution of SPARQL queries on concepts stored in an ontology in SeMFIS, or the translation of an ontology to different syntax formats, e.g. to retrieve the ontology information in OWL Manchester syntax. Similarly, also other communication styles could be implemented in this way, e.g. via REST and JSON. However, the illustrated principles for composing requests would stay the same.

```
CC "Core" GET_ALL_OBJS_WITH_ATTR_VAL
 modelid: (ontologymodel)
 classid: (owlinstanceid)
 attrid: (rdftype)
 val: "Currency (Class) - currency-ontology-model
  (Ontology Model)"
FOR o in: (objids) {
 GET_DATATYPE_PROPERTY_VALUE
  objid: (o)
  dtproperty: "dbp:isoCode"
...
 SET currencyfrom:(dtpropvalue)
 SET currencyto:(targetcurrency)
 SET  soaprequest:(
  "<?xml version=\"1.0\" encoding=\"utf-8\"?>
    <soap:Envelope xmlns:xsi=\"http://www.w3.org/
    ...
     <soap:Body>
      <ConversionRate xmlns=\
      "http://www.webserviceX.NET/\">
       <FromCurrency>" + currencyfrom + "
       </FromCurrency>
       <ToCurrency>" + currencyto + "
       </ToCurrency>
      </ConversionRate>
     </soap:Body>
    </soap:Envelope>")
 CC "AdoScript" GET_TEMP_FILENAME
 SET requestfile:(filename)
 CC "AdoScript" FWRITE
  file: (requestfile)
  text: (soaprequest)
...
```

Program Code 5. Assigning a SOAP Request to an ADOscript Variable and Storing it in a File

The next step is to actually send the request file to the SOAP endpoint over the web. Although ADOscript does not contain implementations of web protocols it-

[6]The service description in WSDL format can be found here `http://www.webservicex.net/CurrencyConvertor.asmx?WSDL` last accessed 14-09-2015

[7]See for a sample instance the DBPedia entry for the Euro currency: `http://dbpedia.org/page/Euro` last accessed 14-09-2015

self, several tools are available that have been specifically designed for this purpose.

In listing 6 we use Wget for Windows in Version 1.11.4[8]. Wget is a non-interactive, command-line utility to retrieve and send files using HTTP, HTTPS, and FTP. It can be called from ADOscript using SYSTEM call statements.

An alternative to this, which may be superior in performance, e.g. by avoiding in-between file operations, would be to write an intermediary component by oneself, e.g. using Java, C++, or Pascal. Especially languages such as Java and C# are well-suited for efficient implementations of web components and offer a large variety of APIs for many different protocols. When providing an intermediary component as a dynamic link library (DLL) it can also be directly called from ADOscript without having to access the operating system. For this purpose, ADOscript offers the CALL operation for calling functions directly in DLLs. However, for most scenarios we have come across so far, calls via SYSTEM and Wget are sufficiently performant and easier to debug for tasks related to semantic-based modeling.

```
SET soapcall: ("
  CMD /c wget.exe http://www.webservicex.net/
   CurrencyConvertor.asmx
   -- post-file=\"" + requestfile + "\"
   -- header=\"Content-Type: text/xml\"
   - O \"" + responsefile + "\"")
SYSTEM (soapcall)
```

Program Code 6. Composing the SOAP Call and Executing it from ADOscript via Wget

As a result of the call via Wget, a SOAP response is received and stored in a file. This file in XML format can be parsed and the result transferred back to ADOscript. In listing 7 we show how this can be accomplished using XMLStarlet in version 1.6.1[9] as another external tool. XMLStarlet is also a command line utility. It provides functions for transforming, querying, validating, and editing XML documents and files. Although it has originally been developed for Unix-based environments, also a Windows version is available. One particular feature are its operations for evaluating XPath queries.

This functionality is used to process the SOAP response message and retrieve the result of the currency conversion. The result is again stored in a file that can be read from ADOscript. The result is finally shown in an infobox UI component that is generated from ADOscript. Again, this could also be realized using a specifically designed DLL or a Java program for improving performance.

```
SET xmlcall: ("
  CMD /c xml.exe sel -N
   ws=http://www.webserviceX.NET/
   -t -v \"//ws:ConversionRateResponse\" "
   + responsefile + " >" + resultfile)
SYSTEM (xmlcall)
CC "AdoScript" FREAD file: (resultfile)
CC "AdoScript" INFOBOX
 ("The current exchange rate for
 " + cur1 + " to " + cur2 + " is: " + text)
```

Program Code 7. Composing the XPath Query, Executing it from ADOscript via XMLStarlet, and Displaying the Result in an Infobox

With the shown code listings the main steps of calling external services from ADOscript could be shown. Of course, for productive applications additional checks of transferred and received data would have to be included. Apart from the tools used here, other options would include for example various tools for interacting with ontologies – e.g. as shown for SeMFIS in [14] for interacting with Protégé and the Jess rule engine on a programmatic level.

For accessing SeMFIS over the web it can be reverted to a built-in SOAP interface. With the code shown in listing 8 the webservice access via SOAP is started or stopped by using a menu item. The menu item definition is part of the standard SeMFIS distribution. The menu entry is thereby added to the Import/Export component ( ) for starting the web service interface on the specified port 1080. Upon the start of the web service, SOAP calls containing ADOscript statements can be made to the platform[10].

A sample SOAP call to SeMFIS is shown in listing 9. At its core it contains ADOscript code that is

[8] See `http://gnuwin32.sourceforge.net/packages/wget.htm` last accessed 31-08-2015

[9] `http://sourceforge.net/projects/xmlstar/` last accessed 31-08-2015

[10] For further information on the web service interface see `http://www.adoxx.org/live/aodxx-web-service`

```
ITEM "~Start Web Service..."
  importexport: "~Web Service Interface"
CC "AdoScript" SERVICE start port: 1080
  output: statusbar
```

Program Code 8. ADOscript Definition for a Web Service Menu Entry in SeMFIS

sent to the *execute* operation of the SOAP endpoint. Thereby some characters in ADOscript code have to be escaped for ensuring compliance with the XML specification. The excerpt of the ADOscript statement shown in the code retrieves all object instances of the class *Model reference* with the identifier stored in the variable *modelid*. It could be part of a larger ADOscript script, e.g. for processing semantic annotations externally. The SOAP response message then contains the identifiers of the objects as specified for the variable *result*.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv=
 "http://schemas.xmlsoap.org/soap/envelope/"
...
 <soapenv:Body>
  <ns1:execute xmlns:ns1="urn:AdoWS">
  <script xsi:type="xsd:string">
   ...
   CC &quot;Core&quot; GET_ALL_OBJS_OF
    _CLASSNAME
    modelid:(modelid)
    classname:&quot;Model reference&quot;
   ...
   SETG result:(objids)
  </script>
  <resultVar xsi:type="xsd:string">
   result
  </resultVar>
  </ns1:execute>
 </soapenv:Body>
</soapenv:Envelope>
```

Program Code 9. Excerpt of a SOAP Message for the SeMFIS SOAP Endpoint with Embedded ADOscript Code in Bold Print

In addition to analyses with ADOscript, SeMFIS also integrates the analysis component of the ADOxx platform (). By using this component queries expressed in AQL (ADOxx Query Language) can be composed and executed. Although this language is currently not as powerful as SQL - e.g. joins are not yet available - it can be used for easily gathering information from the models. An example for a query is given in Figure 3. It shows a query definition targeting all ontology reference elements in a semantic annotation model. The results of these queries can then either be interpreted by users, exported in different file formats such as rtf, csv, or html or accessed programmatically with ADOscript for further processing.

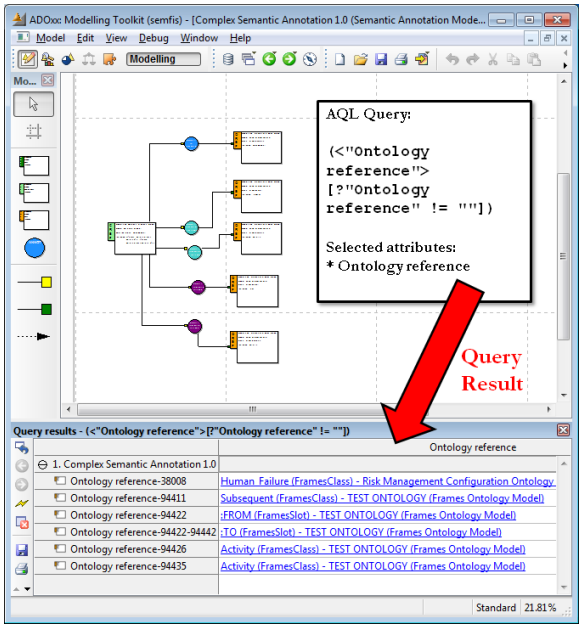

Fig. 3. Example for an AQL Query in SeMFIS retrieving Ontology Reference Elements in a Semantic Annotation Model Instance

3.3. Import and Export Interfaces

SeMFIS provides import and export interfaces for exchanging model information in different file formats. In addition, also an export plugin for the Protégé platform is available. If necessary, these interfaces can also be accessed via ADOscript, e.g. to support additional import/export formats.

3.3.1. Generic XML/ADL Import/Export

The generic XML import/export interface is used to exchange information from arbitrary model types. It is generic in the sense that it does not have to be adapted for specific model types or if modifications are made to a model type. It is therefore well suited for exchanging information with other tools and platforms. Based on the underlying ADOxx platform, SeMFIS also comes

with an integrated Java runtime environment and the Saxon XSLT and XQuery Processor. Thereby, transformations to other XML or file formats can be easily realized. Additionally, via the ADL import/export interface model, information in the proprietary ADOxx-ADL format can be exchanged with other ADOxx-based tools that do not offer an XML interface.

3.3.2. SeMFIS Export Plugin for Protégé

In order to facilitate the exchange of ontology information, a plugin for the Protégé ontology management toolkit has been developed. Protégé has been chosen due to its wide-spread adoption in many fields of science and industry, its large user base and its open accessibility and extensibility [30]. The plugin has recently been adapted in a student project to the Protégé 4.x desktop application. As shown by the screenshot in Figure 4 the plugin is integrated in the Protégé desktop application in the form of a view extension.

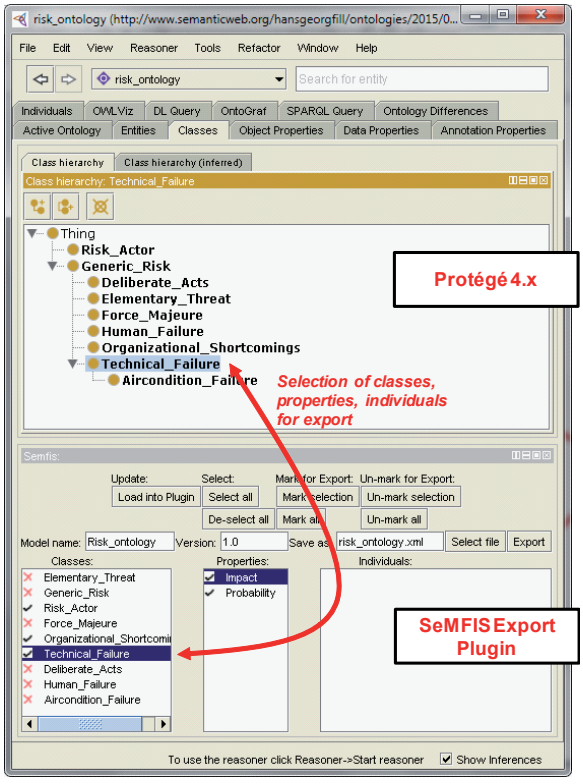

Fig. 4. Screenshot of the SeMFIS Protégé Plugin for Selecting Classes, Properties, and Individuals for Exporting them to SeMFIS

The plugin permits to load classes, properties, and instances from OWL ontologies in the plugin environment. Subsequently it can be chosen which of these elements should be exported to SeMFIS. The selected elements are then stored in a SeMFIS-compatible XML file, which can be directly imported in SeMFIS and visualized in the according OWL ontology model editor. The current implementation of the plugin and the SeMFIS meta model for OWL ontologies only support the major OWL constructs. It is planned however for the future to adapt both the meta model and the plugin, in particular also to support constructs from the OWL2 standard.

4. Architecture of SeMFIS

SeMFIS has been realized using the Microsoft Windows-based ADOxx meta modeling platform. ADOxx is professionally developed by BOC Group, a spin-off of the University of Vienna. It has been on the market for more than fifteen years and has since been used for a large number of research and industry projects [24]. The meta modeling approach of ADOxx is also the basis for the commercial ADONIS business process management toolkit[11], the ADOscore strategic management toolkit [40], and the ADOit enterprise architecture management platform[12], which are well recognized in industry [8]. Since a few years, the ADOxx 1.5 version is available free of charge and can be used for the implementation of academic and industrial modeling methods.

A high level overview of the architecture of the current SeMFIS version is shown in Figure 5. The individual components in the figure are marked as *adapted/configured* components if components from the ADOxx meta modeling platform have been adapted or configured for SeMFIS, as *re-used* components if components from ADOxx have been re-used without modifications for SeMFIS, or as *newly-designed* components if components have been specifically created for SeMFIS. At the bottom rests the repository with the modeling subsystem and a relational database. Although connectors for several databases are available for ADOxx - including Oracle and DB2 - the standard installation of SeMFIS is based on Microsoft SQLServer. The modeling subsystem is a Microsoft Windows application written in C++ and is responsible for translating the information from the meta models and models to a relational database model, for han-

[11] http://www.adonis-community.com/
[12] http://www.adoit-community.com/

dling the persistence of the models in the database, for executing ADOscript statements and for managing the user authentication. If necessary, the ADOxx platform underlying SeMFIS can also be configured for client-server scenarios. Thereby additional features are made available. These include simultaneous multi-user access to the models from several clients including detailed rights administration as well as a more advanced web service access featuring load-balancing mechanisms and automatically generated web service descriptions in WSDL.

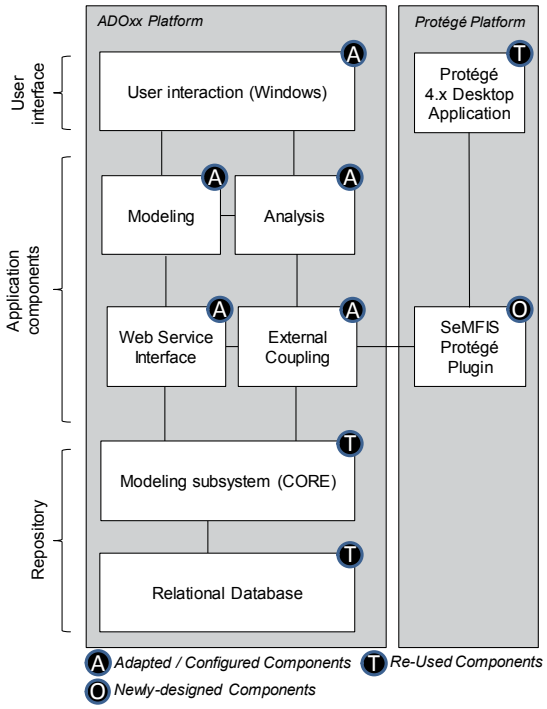

Fig. 5. SeMFIS Architecture based on ADOxx

On top of the repository operate the application components for a. *modeling*, which handles the model editors, b. *analysis*, which provides the AQL query component, c. the *web service interface* that accepts SOAP calls containing ADOscript statements, and d. for the *import/export* of model information via XML/ADL. The Protégé plugin is integrated via the XML interface of the import/export application component.

The user interaction is accomplished through a standard Microsoft Windows desktop user interface. Several aspects of the user interface can be customized using ADOscript, e.g. as shown in the ADOscript examples in section 3.2. This also includes adding functionalities for intercepting certain user actions, e.g. to enforce constraints when creating elements in models, to prevent the modification of certain models, or for triggering the automated execution of complex modification actions.

A particular feature of the SeMFIS architecture is its focus on extensibility and adaptability. As we will discuss in the following section in the context of the use cases, SeMFIS can be easily integrated with other tools and services. Through the customization features provided by ADOxx – see [24] for a detailed discussion – new model types, model elements, and attributes can be added in SeMFIS.

5. Use Cases

SeMFIS has been applied to several use cases and two industry projects in the context of enterprise information systems [21,26,27,12,13,14,15,28]. In the following we will briefly describe two of these use cases to illustrate the possibilities offered by SeMFIS. One use case will be taken from the area of risk management [14] and one from the area of semantic service discovery [28].

In [14] we deployed SeMFIS for the representation, analysis, and simulation of risks in business processes. Based on the semantic annotation approach of SeMFIS, models in a business process modeling language were annotated with concepts from a risk knowledge base. This risk knowledge base was specified in the form of a frames ontology and contained concepts to describe risks and their impact including details on their probability distributions. By using ADOscript together with the XML export interface of SeMFIS, the annotation data was sent to a specifically created mediator component. This mediator component then processed the data by using the Protégé API for frames ontologies. Together with the Jess tab plugin for Protégé, the data was subsequently processed by a Jess rule engine to determine the effects of the assigned risks on the annotated business process elements. These effects were represented in the form of ADOscript statements that were used as input for running simulations on business process models in SeMFIS. The conceptual architecture of this approach is depicted in Figure 6. More details on the technical implementation can be found in [14].

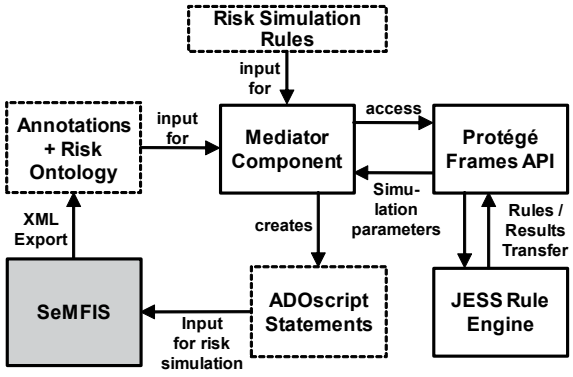

Fig. 6. Conceptual Architecture of Using SeMFIS together with Protégé and the JESS Rule Engine for Risk-based Simulations [14]

By using rules it could be easily defined which types of risks should be analyzed, e.g. based on the type of risk as expressed in the frames ontology. The result of the application of the rules were executable statements in ADOscript that specified how a certain modeling element would be affected by an assigned risk – e.g. that the execution time of an activity in a process is extended when the risk materializes. This information was fed back into the SeMFIS platform via ADOscript and used as input for a business process simulation algorithm. As a result, the effects of risks on business processes could be quantitatively assessed in the simulation runs.

The particular advantage of SeMFIS in this use case was twofold. First, it was not necessary to modify the business process modeling language in order to represent the risks and their effects on business processes. Therefore, none of the existing functionalities working on these models had to be modified nor even considered during implementation. Second, the availability of the ADOscript language and the XML interface permitted to easily interact with the third-party tools. Further details on the use case and in particular the technical architecture can be found in [14].

The second use case is positioned in the area of technology-oriented knowledge management and in particular the semantic discovery of web services [28]. In this use case SeMFIS was integrated in a service-oriented architecture that provided a web-based modeling component via Java applets and a web-based client of the Protégé platform. For this purpose the client-server configuration of ADOxx was enabled. Thereby several instances of SeMFIS services were made accessible via a common WSDL interface that distributed incoming calls through a load balancer service. On the client side, users could access the models stored in SeMFIS via a Java-based web modeller. To establish the link to Protégé, an ontology/model transformation service was added to the architecture. This service resembled the SeMFIS-Protégé plugin described above. Instead of a manual XML import and export of OWL ontologies from Protégé, this service provided SOAP-based interfaces that directly fed the ontology information exported from Protégé to SeMFIS.

Concerning the meta models, the second use case only re-used the OWL ontology model type of SeMFIS. In addition, a model type for representing an extended version of BPMN diagrams and another model type for representing the contents of WSDL files were added – see figure 7. The BPMN diagram was thereby extended with particular operation elements for representing web service calls. With these elements and the XML export interface, BPEL files could be generated that were executable on a BPEL workflow engine. Instead of using the semantic annotation model type provided by SeMFIS, the operation elements in BPMN were directly linked to the WSDL and the OWL ontology model types. The reason was that the BPMN diagram had already been extended so that the advantage of loose coupling via the annotation model type was not necessary. In this way operations in workflows expressed in BPMN could directly reference a web service call. In addition, requirements for such calls could also be described through linkages to the ontology. Similarly, operations in the WSDL could be linked to ontology concepts. A corresponding semantic discovery mechanism written in ADOscript could then propose WSDL operations whose linked ontology concepts matched those of the BPMN operations. In this way, one of the mostly discussed use cases in the area of Semantic Business Process Management could be realized.

From the two use cases presented above three main lessons could be learned. The first lesson learned was that semantic annotations of conceptual models can lead to benefits in terms of processing model information, which go beyond the traditional approaches of formally encoding natural language information of the models. In particular, additional information required for applying analysis and simulation algorithms can be added in this way. This corresponds to similar approaches used in the field of object-oriented programming where additional behavior is added to programs through source code annotations [39]. For the field of

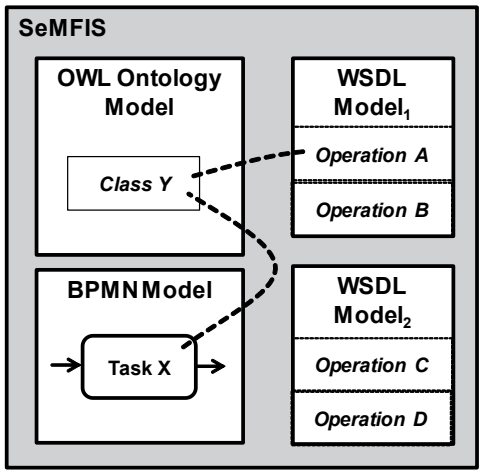

Fig. 7. Illustration of the Concepts Involved in Using SeMFIS for Semantic Discovery of Web Services

modeling this means that additional knowledge is encoded without having to change a modeling language, while at the same time enabling the processing of existing and newly added information.

The second lesson learned was that an important feature of SeMFIS has to be the adaptatbility to different scenarios including different modeling languages and algorithms. Whereas certain key characteristics of semantic annotations such as the semantic annotation model type together with some well-established ontology languages are indispensable for any scenario, the choice of the modeling language or particular analysis and simulation algorithms need to be selectable by the user. With the presented architecture of SeMFIS that is built upon a metamodel-oriented approach, the required flexibility is well achieved.

The third lesson learned was that scenarios for semantic annotations of conceptual models benefit a lot from the availability of web-based interfaces. This not only allows to realized complex analysis scenarios where external services are being called for processing information of semantic annotations and passing back the results to models. Equally important are interfaces for accessing semantic annotations and model information from external systems. The insights gained on these aspects have recently found their way into new approaches for combining services and conceptual models, e.g. for the domain of knowledge-based big data analyses [23] and service composition [25].

6. Discussion and Limitations

With the provision of SeMFIS as an open accessible and extendable platform we believe that we can contribute to the further advancement of approaches in the context of the semantic annotation of conceptual models. SeMFIS is thus positioned as a platform for engineering approaches working at the interface of semi-formal, visual representations in the form of conceptual models and rigid, formally defined ontologies.

With the set of meta models, exchange interfaces, and scripting and analysis mechanisms provided by SeMFIS, a wide variety of future applications may be realized. The set of functionalities that is part of the current version of SeMFIS has evolved in a bottom-up fashion from several research and industrial projects. At the same time many theoretical considerations have found their way into the described concepts, including in particular principles from service orientation such as loose coupling, re-usability, composability, and abstraction. SeMFIS is however not a static tool but rather a starting point for innovative endeavors that want to build upon a solid core that can be adapted to individual needs.

SeMFIS has so far been used in a number of research projects and by a considerable number of users comparable to other tools in the area of semantic web – since 2013, the SeMFIS website reached more than 2.000 downloads. The future potential impact of SeMFIS is also notable due to the fact that it has been built using the ADOxx meta modeling platform. As ADOxx forms the basis for the products developed by the spin-off of the University of Vienna BOC-AG, the functionalities of SeMFIS can be directly added to all BOC products, which are based on the same platform. For its 20-year anniversary BOC-AG reported a user community of 85.000 industry users worldwide and more than 35.000 installations with its clients [2].

Apart from our own research projects, SeMFIS has also been used in several student projects at the University of Vienna. Most recently it was utilized in a course on conceptual models and ontologies at the International Summerschool on Next Generation Enterprise Modeling (NEMO) in 2014 and 2015[13] as well as for two master-level courses on knowledge engineering at the University of Vienna. In these courses the students were instructed about the SeMFIS approach in 90 minute lectures. Each of the students had to com-

[13] See the homepage of the summerschool 2015: `http://www.omilab.org/web/guest/camp2015/`

plete a case study which is publicly available free of charge [18] by using the SeMFIS tool. Subsequently, the students were asked to complete a survey on the usage of SeMFIS. In this survey we were interested in finding out about the ease of use of SeMFIS, whether any features were missing and whether the tool is useful for conducting semantic annotations of conceptual models. In the meantime the survey has also been made publicly available on the SeMFIS website[14]. The participation in the survey was anonymous, not mandatory and did not influence students' grades. In total, 42 students completed the survey, which corresponds to a response rate of about 25% of all students who had participated in the case study. 86% of the respondents found the tool easy to use, 74% found it easy to learn how to interact with the tool, 74% found it well suited for the semantic annotation of conceptual models, 76% agreed that they would recommend the tool to others, and 69% found the tool useful in general. 79% of the students already possessed experience with some other modeling tool. On average the students already had 2.6 years of relevant work experience and most had a computer science or business informatics background. The mostly requested features concerned UML and BPMN as additional modeling languages and a more detailed documentation of the tool's functionalities. As a consequence, BPMN and UML class diagrams had been added to the most recent download version of SeMFIS. Additionally, the documentation on the SeMFIS website had been further expanded regarding video tutorials and the integration of show cases from students' projects for extending SeMFIS.

Regarding its limitations, one important constraint of SeMFIS is that it is only available as a Microsoft Windows application. This results from the fact that the ADOxx platform currently only supports this operating environment. Although this is not a limitation to most users as Windows operating systems dominate the market with a share of about 90% according to recent statistics[15], especially technically adept users may be affected. This limitation may be partly overcome by using the web service interface, but the visual editors cannot be re-used or need to be re-implemented as described above in the second use case. Furthermore, for users with other operating systems, the most common workaround is to run SeMFIS in a virtual machine that hosts Windows. This may not be an optimal solution but gives full access to the SeMFIS functionalities.

Another limitation of SeMFIS is that it does not yet support the most recent OWL2 standard as well as some axioms of the previous OWL standard including certain restriction types and cardinalities. We are however confident to resolve these issues in the oncoming versions of the platform.

7. Conclusion and Outlook

In this paper we have presented SeMFIS – a flexible engineering platform for semantic annotations of conceptual models. SeMFIS provides a set of meta models, exchange interfaces, and scripting and analyis mechanisms for annotating conceptual models with concepts from ontologies. In particular it offers a semantic annotation approach that can be applied to arbitrary modeling languages without requiring their adaptation. SeMFIS has been implemented using the freely available ADOxx platform and is available for free via http://www.omilab.org/web/semfis/. For the future we plan to enhance the coverage of the OWL and OWL2 standards of the meta models provided in SeMFIS. In addition, further functionalities from current research projects are envisaged to be part of future SeMFIS releases. This includes for example the support of social network-based annotations and the integration of natural language processing functionalities [17].

Acknowledgment

The work on SeMFIS has been partially funded by the Austrian Science Fund (FWF) in the course of an Erwin-Schrödinger-Fellowship project conducted at Stanford University under project number J3028-N23.

References

[1] G. Blair, N. Bencomo, and R.B. France. Models@run.time. *IEEE Computer*, 42(10):22–27, 2009. DOI: 10.1109/MC.2009.326.

[2] BOC AG. The Decisive Step Ahead - 20 Years of BOC Group. ; Last accessed: 14-09-2015.

[3] BOC Asset Management GmbH. ADOxx Realization Case: Business Process Model and Notation (BPMN). http://www.adoxx.org/live/bpmn; Last accessed: 21-08-2015.

[14] See https://de.surveymonkey.com/s/2KBSW6H

[15] See the current statistics for Desktop Operating System Market Shares for August 2015 by netmarketshare.com at https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qpcustomd=0

[4] BOC Asset Management GmbH. ADOxx Realization Case: Unified Modelling Language (UML). http://www.adoxx.org/live/uml; Last accessed: 21-08-2015.

[5] BOC Asset Management GmbH. Modelling Language Implementation on ADOxx. http://www.adoxx.org/live/modelling-language-implementation-on-adoxx; Last accessed: 21-08-2015.

[6] D. Bork and H.-G. Fill. Formal Aspects of Enterprise Modeling Methods: A Comparison Framework. In *47th Hawaii International Conference on System Sciences*, pages 3400–3409. IEEE, 2014. DOI: 10.1109/HICSS.2014.422.

[7] M. Born, J. Hoffmann, T. Kaczmarek, M. Kowalkiewicz, I. Markovic, J. Scicluna, I. Weber, and X. Zhou. Semantic Annotation and Composition of Business Processes with Maestro. In S. Bechhofer, M. Hauswirth, J. Hoffmann, and M. Koubarakis, editors, *The Semantic Web: Research and Applications – 5th European Semantic Web Conference*. Springer, 2008. DOI: 10.1007/978-3-540-68234-9_56.

[8] T. DeGennaro, A. Cullen, H. Peyret, and M. Cahill. The Forrester Wave (TM): EA Management Suites, Q2 2013. Technical report, Forrester Research, 2013. ; Last accessed: 21-03-2016.

[9] M. Dimitrov, A. Simov, S. Stein, and M. Konstantinov. A BPMO based Semantic Business Process Modelling Environment. In M. Hepp, K. Hinkelmann, D. Karagiannis, R. Klein, and N. Stojanovic, editors, *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management*, volume 251. CEUR Workshop Proceedings, 2007. http://ceur-ws.org/Vol-251/paper13.pdf; Last accessed: 22-03-2016.

[10] M. Ehrig, A. Koschmider, and A. Oberweis. Measuring Similarity between Semantic Business Process Models. In *4th Asia-Pacific Conference on Conceptual Modelling (APCCM 2007)*, pages 71–80. ACM, 2007.

[11] P. Fettke and P. Loos. Model Driven Architecture (MDA). *Wirtschaftsinformatik*, 45(5):555–559, 2003. DOI: 10.1007/BF03250921.

[12] H.-G. Fill. On the Conceptualization of a Modeling Language for Semantic Model Annotations. In O. Salinesi, C. and Pastor, editor, *Advanced Information Systems Engineering Workshops - CAiSE 2011 International Workshops*, pages 134–148. Springer, 2011. DOI: 10.1007/978-3-642-22056-2_14.

[13] H.-G. Fill. Using Semantically Annotated Models for Supporting Business Process Benchmarking. In M. Grabis, J. and Kirikova, editor, *Perspectives in Business Informatics Research - 10th International Conference, BIR 2011*, volume 90, pages 29–43. Springer, 2011. DOI: 10.1007/978-3-642-24511-4_3.

[14] H.-G. Fill. An Approach for Analyzing the Effects of Risks on Business Processes Using Semantic Annotations. In *European Conference on Information Systems 2012*. AIS, 2012. http://aisel.aisnet.org/ecis2012/111; Last accessed: 22-03-2016.

[15] H.-G. Fill. SeMFIS: A Tool for Managing Semantic Conceptual Models. In H. Kern, J.-P. Tolvanen, and P. Bottoni, editors, *Workshop on Graphical Modeling Language Development*. ECMFA, 2012. http://dsmforum.org/events/GMLD12/papers/Fill.pdf; Last accessed: 22-03-2016.

[16] H.-G. Fill. Using Obfuscating Transformations for Supporting the Sharing and Analysis of Conceptual Models. In S. Robra-Bissantz and D. Mattfeld, editors, *MKWI 2012*. GITO Verlag, 2012. https://eprints.cs.univie.ac.at/3417/; Last accessed: 21-03-2016.

[17] H.-G. Fill. On the Social Network based Semantic Annotation of Conceptual Models. In R. Buchmann, C.V. Kifor, and J. Yu, editors, *7th International Conference on Knowledge Science, Engineering and Management*, pages 138–149. Springer, 2014. DOI: 10.1007/978-3-319-12096-6_13.

[18] H.-G. Fill. Enabling Risk Analysis in Conceptual Models by Using Semantic Annotations - Case Study for Semantic-Based Modeling, 2015. URL: ; Last accessed: 21-08-2015.

[19] H.-G. Fill. SeMFIS Developer-Tutorial 3: Extending SeMFIS with Classes, Relationclasses, and Model Types, 2015. URL: ; Last accessed: 21-08-2015.

[20] H.-G. Fill. SeMFIS Developer-Tutorial 4: Adding Classes of New Model Types to the Model Reference Class in the Semantic Annotation Model Type, 2015. URL: ; Last accessed: 21-08-2015.

[21] H.-G. Fill and P. Burzynski. Integrating Ontology Models and Conceptual Models using a Meta Modeling Approach. In *11th International Protégé Conference*, 2009. ; Last accessed: 21-03-2016.

[22] H.-G. Fill, A. Gericke, D. Karagiannis, and R. Winter. Modeling for Integrated Enterprise Balancing (in German). *Business and Information Systems Engineering*, 49(6):419–429, 2007. DOI: 10.1007/s11576-007-0094-6.

[23] H.-G. Fill and F. Johannsen. A Knowledge Perspective on Big Data by Joining Enterprise Modeling and Data Analyses. In *Proceedings of the 49th Hawaii International Conference on System Sciences*, pages 4052–4061. IEEE, 2016. DOI: 10.1109/HICSS.2016.503.

[24] H.-G. Fill and D. Karagiannis. On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform. *Enterprise Modelling and Information Systems Architectures*, 8(1):4–25, 2013.

[25] H.-G. Fill, D. Karagiannis, and C. Lichka. Integration of Conceptual Models and Data Services Using Metamodeling. In *19th International Enterprise Distributed Object Computing Workshop*. IEEE, 2015. DOI: 10.1109/EDOCW.2015.35.

[26] H.-G. Fill and I. Reischl. An Approach for Managing Clinical Trial Applications using Semantic Information Models. In S. Rinderle-Ma, S. Sadiq, and F. Leymann, editors, *Business Process Management Workshops*, pages 581–592, Ulm, Germany, 2009. Springer. DOI: 10.1007/978-3-642-12186-9_56.

[27] H.-G. Fill and I. Reischl. Stepwise Semantic Enrichment in Health-related Public Management by Using Semantic Information Models. In S. Smolnik, F. Teuteberg, and O. Thomas, editors, *Semantic Technologies for Business and Information Systems Engineering: Concepts and Applications*, volume 195–212. IGI Press, 2011. DOI: 10.4018/978-1-60960-126-3.ch010.

[28] H.-G. Fill, D. Schremser, and D. Karagiannis. A Generic Approach for the Semantic Annotation of Conceptual Models using a Service-oriented Architecture. *International Journal of Knowledge Management*, 9(1):76–88, 2013. DOI: 10.4018/jkm.2013010105.

[29] P.R. Fillottrani, E. Franconi, and S. Tessaris. The ICOM 3.0 Intelligent Conceptual Modelling tool and methodology. *Semantic Web - Interoperability, Usability, Applicability*, 3(3):293–306, 2012. DOI: 10.3233/SW-2011-0038.

[30] J.H. Gennari, M. A. Musen, R.W. Fergerson, W.E. Grosso, M. Crubezy, H. Eriksson, N.F. Noy, and S.W. Tu. The evolu-

tion of Protégé: an environment for knowledge-based systems development. *International Journal of Human-Computer Studies*, 58:89–123, 2003. DOI: 10.1016/S1071-5819(02)00127-1.

[31] T. Gruber. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220, 1993. DOI: 10.1006/knac.1993.1008.

[32] D. Harel and B. Rumpe. Modeling Languages: Syntax, Semantics and All That Stuff - Part I: The Basic Stuff. Technical Report MCS00-16, The Weizmann Institute of Science, August 22 2000.

[33] M. Hepp, F. Leymann, J. Domingue, A. Wahler, and D. Fensel. Semantic business process management: A vision towards using semantic web services for business process management. In *International Conference on e-Business Engineering, 2005. ICEBE 2005*, pages 535–540. IEEE, 2005. DOI: 10.1109/ICEBE.2005.110.

[34] J. Herbst, S. Junginger, and H. Kühn. Simulation in Financial Services with the Business Process Management System ADONIS. In W. Hahn and A. Lehmann, editors, *Proceedings of the 9th European Simulation Symposium (ESS'97)*, pages 491–495. Society for Computer Simulation, 1997.

[35] P. Höfferer. Achieving Business Process Model Interoperability Using Metamodels and Ontologies. In *European Conference on Information Systems*, pages 1620–1631. AIS, 2007. http://aisel.aisnet.org/ecis2007/174; Last accessed: 21-03-2016.

[36] S. Joksimovic, J. Jovanovic, D. Gasevic, A. Zouaq, and Z. Jeremic. An empirical evaluation of ontology-based semantic annotators. In *Proceedings of the seventh international conference on Knowledge capture*, pages 109–112. ACM, 2013. DOI: 10.1145/2479832.2479855.

[37] X. Juicheng, B. Zhaoyang, A.J. Berre, and O.C. Brovig. Model Driven Interoperability through Semantic Annotations using SoaML and ODM. In N. Bakhtadze and A. Dolgui, editors, *Information Control Problems in Manufacturing*, volume 13, pages 650–655, 2009. DOI: 10.3182/20090603-3-RU-2001.00108.

[38] R. H. Kaschek. On the evolution of conceptual modeling. In R.H. Delcambre, L. and Kaschek and H.C. Mayr, editors, *Dagstuhl Seminar Proceedings*, volume 08181. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2008. ; Last accessed: 21-03-2016.

[39] H. Krahn and B. Rumpe. Towards Enabling Architectural Refactorings through Source Code Annotations. In H.C. Mayr and R. Breu, editors, *Modellierung 2006*, volume 82, pages 203–212. GI-LNI, 2006. http://arxiv.org/abs/1409.6612; Last accessed: 21-03-2016.

[40] C. Lichka, H. Kühn, and D. Karagiannis. IT-gestützte Balanced Scorecard. *Wisu*, 7:915–918, 2002.

[41] Y. Lin. *Semantic Annotation for Process Models: Facilitating Process Knowledge Management via Semantic Interoperability*. PhD thesis, Norwegian University of Science and Technology, 2008. http://www.idi.ntnu.no/research/doctor_theses/yunl.pdf; Last accessed: 22-03-2016.

[42] J. Mylopoulos. Conceptual Modeling and Telos. In P. Loucopoulos and R. Zicari, editors, *Conceptual Modelling, Databases and CASE: An Integrated View of Information Systems Development*, pages 49–68. Wiley, 1992.

[43] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W.R. Swartout. Enabling technology for knowledge sharing. *AI Magazine*, 12(3):36–56, 1991. DOI: 10.1609/aimag.v12i3.902.

[44] Object Management Group OMG. Unified Modeling Language (UML), Infrastructure, V2.1.2, 2007. http://www.omg.org/spec/UML/2.1.2/Infrastructure/PDF/; Last accessed: 01-03-2011.

[45] Object Management Group (OMG). Business Process Model and Notation (BPMN) Version 2.0, 2011. http://www.omg.org/spec/BPMN/2.0/PDF; Last accessed: 01-03-2011.

[46] M. Rosemann. Potential pitfalls of process modeling: part B. *Business Process Management Journal*, 12(3):377–384, 2006. DOI: 10.1108/14637150610668024.

[47] Softtech Inc. Integrated Computer-Aided Manufacturing (ICAM) - Architecture Part II - Volume IV Function Modeling Manual ($IDEF_0$), 1981. http://www.dtic.mil/dtic/tr/fulltext/u2/b062457.pdf; Last accessed: 04-09-2012.

[48] S. Stein, C. Stamber, and M. El Kharbili. ARIS for Semantic Business Process Management. In D. Ardagna, M. Mecella, and J. Yang, editors, *Business Process Management Workshops*, pages 498–509. Springer, 2009. DOI: 10.1007/978-3-642-00328-8_50.

[49] V. Uren, P. Cimiano, J. Iria, S. Handschuh, M. Vargas-Vera, E. Motta, and F. Ciravegna. Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4:14–28, 2006. DOI: 10.1016/j.websem.2005.10.002.

[50] W. M. P. Van der Aalst. Formalization and Verification of Event-driven Process Chains. *Information and Software Technology*, 41(10):639–650, 1999. DOI: 10.1016/S0950-5849(99)00016-6.

[51] N. Visic. MM-DSL: An EBNF Specification Version 1.0. Technical report, University of Vienna, Faculty of Computer Science, 2013. http://www.omilab.org/web/guest/mm-dsl; Last accessed: 21-08-2015.

[52] N. Visic, H.-G. Fill, R.A. Buchmann, and D. Karagiannis. A Domain-specific Language for Modeling Method Definition: from Requirements to Grammar. In *International Conference on Research Challenges in Information Science*, pages 286–297. IEEE, 2015. DOI: 10.1109/RCIS.2015.7128889.

[53] Y. Wand and R. Weber. Research Commentary: Information Systems and Conceptual Modeling - A Research Agenda. *Information Systems Research*, 13(4):363–376, 2002. DOI: 10.1287/isre.13.4.363.69.

[54] E. Yu, P. Giorgini, N. Maiden, and J. Mylopoulos. *Social Modeling for Requirements Engineering*. MIT Press, 2011.

[55] W. Yuxin and L. Hongyu. Adding Semantic Annotation to UML Class Diagram. *International Conference on Computer Application and System Modeling*, pages V9–187–V9–190, 2010. DOI: 10.1109/ICCASM.2010.5623055.