



---

## D2.4.1 Metadata Annotation Tools. Components version 1.

---

Moritz Weiten

Mika Meier-Collin, Jürgen Angele (all ontoprise GmbH)

### **Abstract**

This document describes the components resulting from task 2.4 of the Metadata Generation work package. The components allow the semantic enrichment of existing data resources. Two components have been created: one for the annotation of semi-structured data in form of spreadsheet-annotations and one for the manual and semi-automatic annotation of MS Word®-documents. The latter actually encapsulates existing Information Extraction technology.

The components interoperate with the OntoStudio® ontology engineering workbench. OntoStudio® provides the model-data used for annotations (on a schema-level). The Spreadsheet-Plugin stores annotation metadata in the ontology, while the text-based annotations are stored within the Word®-documents. The advantages and disadvantages as well as the use-cases are briefly discussed.

Both tools work in a very different way, serving different kinds of use-cases. Two simple scenarios illustrate in a step-by-step manner how the tools are actually applied. Limitations and future extensions are discussed as well.

Keyword list: annotation, ontologies, spreadsheets, information extraction

WP2 Metadata Generation

Prototype

PU

Contractual date of delivery

Actual date of delivery

2005-12-31

2006-01-26

### CHANGES

<b>Version</b>	<b>Date</b>	<b>Author</b>	<b>Changes</b>
0.3	15.12.05	Moritz Weiten	Initial Version
0.5	03.01.06	Moritz Weiten	Included Input from Mika Maier-Collin regarding OntoOffice IE
0.7	20.01	Moritz Weiten	Pre-Final version
0.8	25.01.	Moritz Weiten	Included Paul's review feedback except for the OBIE-related issues.
0.9	26.01	Moritz Weiten	Included some comments about the relation to OBIE and plans to support it

## SEKT Consortium

This document is part of a research project partially funded by the IST Programme of the Commission of the European Communities as project number IST-2003-506826.

### **British Telecommunications plc.**

Orion 5/12, Adastral Park  
Ipswich IP5 3RE  
UK  
Tel: +44 1473 609583, Fax: +44 1473 609832  
Contact person: John Davies  
E-mail: [john.nj.davies@bt.com](mailto:john.nj.davies@bt.com)

### **Empolis GmbH**

Europaallee 10  
67657 Kaiserslautern  
Germany  
Tel: +49 631 303 5540  
Fax: +49 631 303 5507  
Contact person: Ralph Traphöner  
E-mail: [ralph.traphoener@empolis.com](mailto:ralph.traphoener@empolis.com)

### **Jozef Stefan Institute**

Jamova 39  
1000 Ljubljana  
Slovenia  
Tel: +386 1 4773 778, Fax: +386 1 4251 038  
Contact person: Marko Grobelnik  
E-mail: [marko.grobelnik@ijs.si](mailto:marko.grobelnik@ijs.si)

### **University of Karlsruhe, Institute AIFB**

Englerstr. 28  
D-76128 Karlsruhe  
Germany  
Tel: +49 721 608 6592  
Fax: +49 721 608 6580  
Contact person: York Sure  
E-mail: [sure@aifb.uni-karlsruhe.de](mailto:sure@aifb.uni-karlsruhe.de)

### **University of Sheffield**

Department of Computer Science  
Regent Court, 211 Portobello St.  
Sheffield S1 4DP  
UK  
Tel: +44 114 222 1891  
Fax: +44 114 222 1810  
Contact person: Hamish Cunningham  
E-mail: [hamish@dcs.shef.ac.uk](mailto:hamish@dcs.shef.ac.uk)

### **University of Innsbruck**

Institute of Computer Science  
Techikerstraße 13  
6020 Innsbruck  
Austria  
Tel: +43 512 507 6475  
Fax: +43 512 507 9872  
Contact person: Jos de Bruijn  
E-mail: [jos.de-bruijn@deri.ie](mailto:jos.de-bruijn@deri.ie)

### **Intelligent Software Components S.A.**

Pedro de Valdivia, 10  
28006  
Madrid  
Spain  
Tel: +34 913 349 797  
Fax: +49 34 913 349 799  
Contact person: Richard Benjamins  
E-mail: [rbenjamins@isoco.com](mailto:rbenjamins@isoco.com)

## D2.4.1 / Metadata Annotation Tools. Components version 1.

### **Kea-pro GmbH**

Tal

6464 Springen

Switzerland

Tel: +41 41 879 00

Fax: 41 41 879 00 13

Contact person: Tom Bösser

E-mail: [tb@keapro.net](mailto:tb@keapro.net)

Contact person: Pompeu Casanovas Romeu

E-mail: [pompeu.casanovas@uab.es](mailto:pompeu.casanovas@uab.es)

### **Siemens Business Services GmbH & Co. OHG**

Otto-Hahn-Ring 6

81739 Munich

Germany

Contact person: Dirk Ramhorst

Tel: +49 (89)63640225; Fax: +49 89 63640233

Email: [Dirk.Ramhorst@siemens.com](mailto:Dirk.Ramhorst@siemens.com)

### **Ontoprise GmbH**

Amalienbadstr. 36

76227 Karlsruhe

Germany

Tel: +49 721 50980912

Fax: +49 721 50980911

Contact person: Hans-Peter Schnurr

E-mail: [schnurr@ontoprise.de](mailto:schnurr@ontoprise.de)

### **Sirma Group Corp., Ontotext Lab**

135 Tsarigradsko Shose

Sofia 1784

Bulgaria

Tel: +359 2 9768 303, Fax: +359 2 9768 311

Contact person: Atanas Kiryakov

E-mail: [naso@sirma.bg](mailto:naso@sirma.bg)

### **Vrije Universiteit Amsterdam (VUA)**

Department of Computer Sciences

De Boelelaan 1081a

1081 HV Amsterdam

The Netherlands

Tel: +31 20 444 7731, Fax: +31 84 221 4294

Contact person: Frank van Harmelen

E-mail: [frank.van.harmelen@cs.vu.nl](mailto:frank.van.harmelen@cs.vu.nl)

### **Universitat Autònoma de Barcelona**

Edifici B, Campus de la UAB

08193 Bellaterra (Cerdanyola del Vallès)

Barcelona

Spain

Tel: +34 93 581 22 35, Fax: +34 93 581 29 88

## **Executive Summary**

A large part of information available in the Web but also in the intranets of companies is unstructured or semi-structured, making it hard for end-users to find relevant information. One of the main objectives of the SEKT-project is to overcome the limitations of common search-engines but also to provide a whole infrastructure for knowledge-based applications. There are different technologies available to enrich or use existing information sources based on manual and automatic processes. The application of those technologies is more or less complex for users with a basic knowledge of semantic technologies.

This document describes end-user tools which make use of technologies for annotation, information extraction and ontology management. Those tools support the authoring of Documents, the semi-automatic Annotation and (Semi-)Structured Data Annotation based on the MS Office®-platform. The spreadsheet annotation tool is realized in form of an integrated OntoStudio®-plugin whereas the text-based annotation tool works as MS Word®-plugin (exchanging data with OntoStudio®). Both tools interoperate with the OntoStudio® ontology engineering environment. In both cases OntoStudio® provides data on the schema level, e.g. to allow users to select the concepts used for manual annotations. The spreadsheet-plugin also stores the annotation metadata in OntoStudio®, while OntoOffice® IE stores instance data as a result of the annotation process. On a technical level the spreadsheet-plugin is directly coupled with OntoStudio® via the datamodel API (to create and manage ontology data). OntoOffice® IE is connected to OntoStudio via socket connection.

Two simple scenarios illustrate how existing MS Office®-documents can be enriched or exploited as a resource by the use of semantic technologies. Spreadsheets are a common tool to structure resources with high flexibility according to implicit semantics, partially reflected by table-layouts. The first scenario discussed in this document shows, how spreadsheet-tables can be annotated in order to interpret them as a resource for instance data. This enables the enrichment of such tables e.g. in form of “personal knowledge bases” such as a list of favourite journal references. The second scenario shows how text-data can be used to feed a knowledge model although users (providing text) are not forced into detailed structured forms. It illustrates how MS Word can be used for the training of Information Extraction tools and automatic annotation.

Finally both components have limitations. In the current version OntoOffice® IE a single document is associated with a single instance (of a defined concept). This is sufficient for the scenario described in this document but drastically limits the use-cases. Future versions of the plugin will allow the annotation with multiple instances. Limitations with regard to the spreadsheet-plugin concern the support for complex table layouts. However, the room for extensions in this case is limited since the whole complexity of annotation- and IE-technology can not simply be encapsulated by end-

#### D2.4.1 Metadata Annotation Tools. Components version 1.

user tools. In the case of the spreadsheet a graphical user interface would have to be based on a complex workflow and a visualization that is hard to follow due to the fact that the complex annotation model has to be reflected (e.g. visualizing the paths, through which table cells are interpreted as described by ).

## Contents

<b>SEKT Consortium.....</b>	<b>3</b>
<b>Executive Summary .....</b>	<b>5</b>
<b>Contents.....</b>	<b>7</b>
<b>1 Introduction.....</b>	<b>8</b>
<b>2 Annotation Components .....</b>	<b>10</b>
2.1 Spreadsheet Annotations with the Spreadsheet-Plugin .....	12
2.1.1 Related Work.....	12
2.1.2 Semantics of a spreadsheet table annotation .....	13
2.1.3 Scenario .....	15
2.1.4 Preparing the Annotation Process.....	17
2.1.5 Annotating an open spreadsheet .....	19
2.1.6 Testing the Annotations .....	21
2.1.7 Modifications.....	25
2.2 Text-based Annotations with OntoOffice® IE .....	26
2.2.1 Annotation Model.....	27
2.2.2 Scenario .....	28
2.2.3 Preparing the Annotation Process.....	30
2.2.4 Annotating a Document .....	31
2.2.5 Generating Training Data.....	33
2.2.6 Automatic Annotation Process .....	33
2.2.7 Storing Instance Data .....	33
<b>3 Discussion and Outlook.....</b>	<b>34</b>
<b>4 Bibliography and references .....</b>	<b>36</b>

## 1 Introduction

A large part of information available in the Web but also in the intranets of companies is unstructured or semi-structured, making it hard for end-users to find relevant information. One of the main objectives of the SEKT-project is to overcome the limitations of common search-engines but also to provide a whole infrastructure for knowledge-based applications.

However, even though semantic technologies offer the capabilities for efficient information retrieval and knowledge-based applications, unstructured and poorly structured resources will always play an important role. The semantic enrichment of resources through annotations provides a solution to this problem. Annotations provide the metadata necessary to link existing sources with rich and expressive knowledge models, supporting precise domain-specific searches. Those annotations can be performed (i) manually, (ii) automatically or (iii) semi-automatically; (ii) and (iii) are objectives of Information Extraction (IE).

IE can provide support for document annotation by the unsupervised extraction of information or in form of a semi-automatic process, e.g. via information highlighting or generation of proposals for annotations. There is another important aspect in addition to the reuse of existing information sources: The generation of machine-readable content with reasonable effort. Machine-readable content, i.e. content that can be *interpreted* by a machine is one of the basic aspects of Semantic Technologies. Human users would have to be familiar with a representation form with well-defined semantics in order to create this machine readable-content, such as a common ontology language. This requires the right know-how or a significant amount of training. Users of semantic technologies can not in general be expected to be logic experts. If they are to provide the contents which form the base for knowledge-based applications, efficient means of knowledge-acquisition are crucial. Those should be as close to the usual output of typical business processes as possible, avoiding complex interfaces, etc. IE does in addition focus on the efficiency when large amounts of data need to be handled. By the use of machine-learning algorithms, large sets of documents can be annotated in an efficient way.

A common strategy for IE is the so called “mixed-initiative learning” (see [Li2004]). It is based on an iterative process involving manual annotations and a learning algorithm. The latter is trained with a set of annotated text-documents. In the following phase, the algorithm is applied to a new set of documents. Manual corrections of the annotation-results lead to further improvement. Those steps are then repeated until quality of the automatic annotations meets the user’s requirements. The goal of the end-user component for IE described in this document is to provide an end-user tool that encapsulates the functionality of IE-tools. The layman should actually only be given some basic information about the algorithms he can use on his documents without assuming in-depth knowledge of IE methods.

#### D2.4.1 Metadata Annotation Tools. Components version 1.

With respect to annotations of semi-structured data-sources the goal was to provide a graphical user interface that allows users to create metadata for the dynamic access to the sources without knowing about the details of the formal representation and interpretation of those annotations.

Within SEKT and task 2.4 this includes two basic categories of information resource: unstructured resources in form of text documents and semi-structured resources in form of spreadsheet tables. For the latter, capabilities to make the implicit structure and its intended semantics explicit have to be provided. This is not a trivial task, since spreadsheet applications provide limited explicit structuring, compared for example to databases. For text-based resources, natural language processing capabilities are required to handle them efficiently.

This document describes different types of tools (in the form of two basic components) which can be characterized according to the following functionalities:

- § **Authoring of Documents** supporting the generation of meta data during the creation of business documents as part of typical business tasks;
- § **Semi-automatic Annotation** by connecting available technology for manual annotation with IE to reduce the effort for manual annotation;
- § **(Semi-)Structured Data Annotation** to markup dynamically created content, here in the form of spreadsheets.

OntoOffice® IE satisfies to the first and the second category. It allows to generate meta data from within the application. The spreadsheet-plugin belongs to the last category. Both components are described in the next sections, while the focus is on the usage of the tools.

## 2 Annotation Components

The objective of task 2.4 is to provide end-user tools which

- § provide simple graphical user interfaces to sophisticated technology for annotation and information extraction,
- § have a flat learning-curve,
- § support existing tools in typical office environments,
- § can be adapted to incorporate the latest developments of annotation tools.

Therefore the development of annotation components as part of task 2.4 does not include the “backend technology” but the client- or end-user tools.

There are two separate components for the annotation of resources, both working together with the OntoStudio® ontology engineering tool.

### *The Spreadsheet-Plugin for spreadsheet-annotations*

This plugin covers semi-structured resources in form of spreadsheets, based on a semantic interpretation of table structures. It is implemented as a pure Eclipse<sup>1</sup>-/OntoStudio®-Plugin and can therefore only be used as part of the ontology engineering platform. Consequently the targeted end-users are knowledge engineers and domain experts with a basic know-how of semantic technologies but no in-depth knowledge of semantic representation languages. The annotations created by this component are represented internally as rules. Based on those rules annotated tables can be used a source of instance data.

### *The OntoOffice® IE-Plugin:*

The OntoOffice® is based on the MS Office® platform. MS Office® was chosen since it is the most widespread office-environment and provides a rich API.<sup>2</sup> OntoOffice® IE allows to manually annotate documents, store annotations as training data and test the automatic annotation. The annotations are stored within the document. The OntoOffice® IE operates with the OntoStudio® ontology engineering platform as ontology provider. It can also used with the OntoBroker® ontology-server instead. A typical usage scenario would be for example to open the PROTON-ontology from within OntoStudio®, start OntoOffice® IE, open a PROTON-related document and create for example annotations related to (instances of) persons.

---

<sup>1</sup> See [www.eclipse.org](http://www.eclipse.org)

<sup>2</sup> In principle it would be possible to support for example the OpenOffice-environment, since it also provides an API with the required functionality. However, it was decided to first of all support the most popular tool and gain experiences with the approach.

#### D2.4.1 Metadata Annotation Tools. Components version 1.

However, at the end of this document it will be clear that this scenario will be sufficiently supported only with the coming version of OntoOffice® IE since the current prototype has some basic limitations.

The goal of OntoOffice® is to provide an end-user tool that is embedded within their word processing-application in order to integrate the generation of metadata into everyday business activities. Another design-goal was to keep OntoOffice® IE simple and as (IE-)implementation –independent as possible. It currently uses the Amilcare tool as “IE-backend” (see [Cira03]). This IE system is based on an algorithm that falls into the class of Wrapper Induction Systems. It can be run in different modes (such as training, test and production). OntoOffice® IE calls the system in the appropriate mode depending on the action triggered by the user.

In future OntoOffice® IE will also have a binding to the OBIE-system, which is based on SVM-algorithms. Both systems are based on the GATE-platform. Users will be enabled to choose the “backend” that seems to fit their needs in the best way. The integration of IE-functionality within their word-processing tool will encourage them to gain experiences with the technology.

While both tools support the use of existing resources, they work in a very different way:

- § The Spreadsheet-Plugin stores the annotation meta-data in the ontology, while OntoOffice® IE stores it within the document. Storing the meta-data within the document has the following advantages: It makes it easier to deal with certain changes (e.g. if a section is deleted). It allows to work within the “natural environment” of the document where it appears in the way users would expect it (keeping the original layout, displaying non-textual elements, etc.). For word-documents, the latter aspect was considered to be quite important. However, for Spreadsheets graphical components are available which have a very similar “look and feel” like spreadsheet-applications. Storing the metadata outside the spreadsheet makes it possible to annotate write-protected files. Since the spreadsheet-plugin accesses spreadsheets dynamically (see next point). This is a significant advantage, since users can provide their own spreadsheet as an information resource to others. An additional advantage is that several spreadsheet formats can be supported in future, without changing the user interfaces and the basic API.
- § The Spreadsheet-Plugin accesses spreadsheets dynamically. Whenever instance data is requested, the annotation rules are evaluated and instance data is generated out of the actual spreadsheet table. Creating instance data with OntoOffice® IE is based on a batch-process.

## 2.1 Spreadsheet Annotations with the Spreadsheet-Plugin

In many cases MS Excel is used for database-like applications, for example for balances, parts lists, etc. For some types of spreadsheets an added value could be created by “semantic adaptors” (or connectors) that provide access to the spreadsheet data as a source of instance data of a given model, similar to the existing database-integration (schema-import of OntoStudio). MS Excel is seen as an important factory for many retrieval problems, since the amount of data stored in Excel-spreadsheets in typical organizations or companies is tremendous.

However, the access to spreadsheets based on a model is not as straightforward as in the case of XML-schemas or relational databases due to the fact that spreadsheets do not rely on a well-defined, distinct schema. This has important implications on the creation of adaptors. A simple example of a “troublesome” structure would be joined table-cells for example. Another example are related tables without “physical” connection, like a table containing measured or calculated data (raw data) and another table on the same spreadsheet containing meta-information (such as conditions under which the data was created). The relation of the tables is obvious to a human interpreter with appropriate background-knowledge, but it is not explicitly stored (not to talk about a definite, machine-readable relation). The same holds for links between tables, which are not based on a defined key-mechanism (as in relational databases).

Therefore the creation of spreadsheet-adapters will focus on simple table structures, suitable for mappings onto ontologies. It will always involve manual interaction, supported by semi-automated processes. Similar to the annotation-process for text-documents, users will explicitly define the relation of the given spreadsheet-structure and the ontology.

### 2.1.1 *Related Work*

The problem of a (semantic) interpretation of tables with arbitrary structures has been discussed by various authors, e.g. [Zani2004]. The tool described here is much related to the work described in [Pivk2005]. The authors have developed the TARTAR system, which has its focus on web-resources and an automatic transformation process. The approach is based on the F-Logic language as target representation (see [Kife1997]). The authors deal with complex table structures, trying to fully exploit the expressivity of the target language. This includes for example the use of function symbols for certain structures.

The differences to the approach underlying the Spreadsheet-Plugin are:

- § The focus of the annotation plugin is not on automatic annotation.

Even though the goal is to include automatic annotation capabilities in future versions, this will be limited to the generation of suggestions for annotations. [Pivk2005] describes a transformation process, while the Spreadsheet-Plugin is based on a live-access to the annotated table. In the latter case, corrections to

#### D2.4.1 Metadata Annotation Tools. Components version 1.

automatic interpretation of the table structure (once they are supported) will have to be made at “design time”, while the actual data is access during “runtime”. There is no way to correct the resulting instance data directly. The assumption is that an annotation schema is valid once it has been saved, since the annotation plugin does not directly produce instance data but rules<sup>3</sup>.

§ TARTAR supports rather complex table structures while the Spreadsheet-Plugin has its focus on “growing tables”.

The authors of the TARTAR system have shown how complex table structures can be transformed into F-Logic in a step-wise process. Those complex structures are currently not supported by the annotation plugin and only a subset will be supported in future. The assumption of the annotation plugin is, that a table can grow (with respect to the actual data, excluding metadata like headers) without making it necessary to adapt the annotation schema.

§ The annotation plugin does currently not exploit the capabilities of the underlying representation language

The use of function symbols as described in [Pivk2005] allow a redundant-free interpretation of nested column headers for example.

##### 2.1.2 *Semantics of a spreadsheet table annotation*

A detailed discussion of the semantics of tables is given in [Pivk2005]. The authors describe different aspects of the analysis of tables, such as graphical, physical, structural, functional and semantic aspects. Structural aspects concern the organization of cells (how to navigate within the table). Functional aspects relate to the function of table-areas (differentiation between cells containing “actual data” readers are interested in and cells containing path information to retrieve that data). Semantic aspects relate to the interpretation of the table in terms of a model which allows making statements about what is true with respect to that model. It captures different aspects, such as what the meaning of cell contents is and how the table structure is to be interpreted.

In [Pivk2005] a general classification of table-layouts is given. The authors identified three different categories: one-dimensional, two-dimensional and complex tables. Currently the Spreadsheet-Plugin covers only one-dimensional layouts. This excludes for example nested headers. A support of more complex layouts is targeted in future versions, based on an extension of the annotation of header-cells. However, it has not yet been decided which model should be used for this task. The use of function symbols efficiently exploits the expressivity of the target representation, but it limits the interoperability of the annotation schema to tools supporting this modelling primitive.

---

<sup>3</sup> A transformation could be realized via materialization of the rules.

#### D2.4.1 Metadata Annotation Tools. Components version 1.

Figure 1 illustrates how a one-dimensional maps onto an ontology in the Spreadsheet-Plugin. Ranges of spreadsheets (tables) map onto a single concept, meaning that the range is a resource of instance data related to the concept. A single row (or column, depending on the orientation) in the table contains the data for a single instance of the concept. Note that the table might not cover *all* the instance data. The ontology can in principle contain additional instance data.

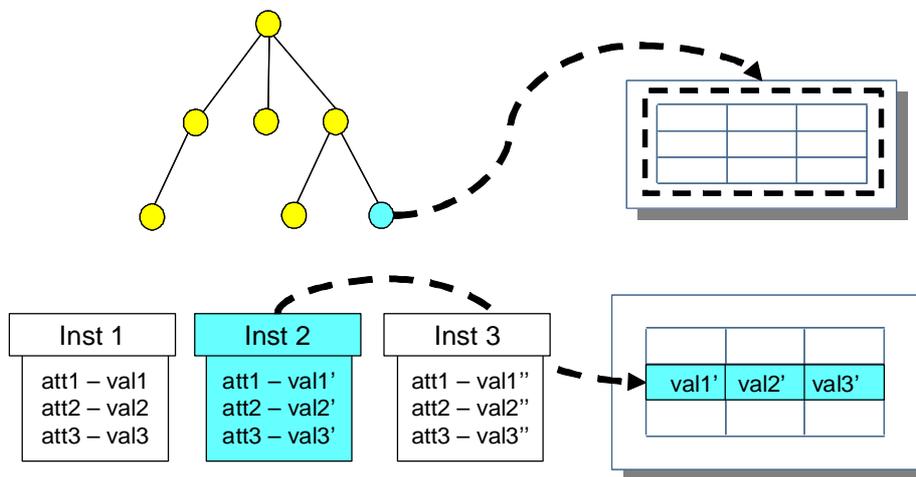
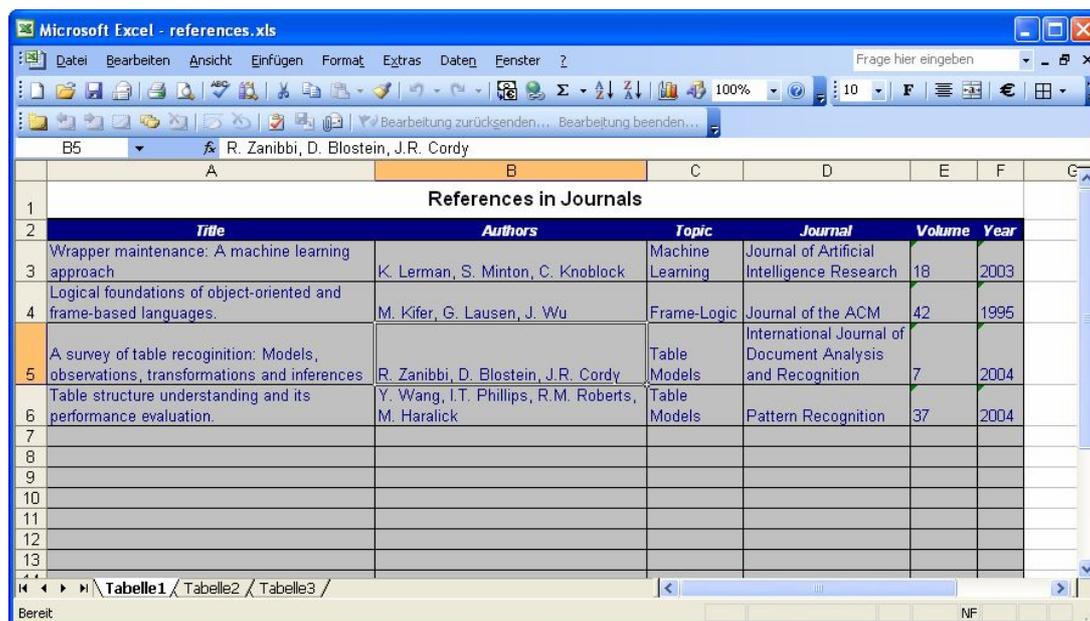


Fig. 1: Simple illustration of the semantics of a spreadsheet annotation

The following section contains an example which describes the usage of the Spreadsheet-Plugin but also explains how an annotated table is interpreted.

## D2.4.1 Metadata Annotation Tools. Components version 1.

### 2.1.3 Scenario



	A	B	C	D	E	F	G
1	References in Journals						
2	<b>Title</b>	<b>Authors</b>	<b>Topic</b>	<b>Journal</b>	<b>Volume</b>	<b>Year</b>	
3	Wrapper maintenance: A machine learning approach	K. Lerman, S. Minton, C. Knoblock	Machine Learning	Journal of Artificial Intelligence Research	18	2003	
4	Logical foundations of object-oriented and frame-based languages.	M. Kifer, G. Lausen, J. Wu	Frame-Logic	Journal of the ACM	42	1995	
5	A survey of table recognition: Models, observations, transformations and inferences	R. Zanibbi, D. Blostein, J.R. Cordy	Table Models	International Journal of Document Analysis and Recognition	7	2004	
6	Table structure understanding and its performance evaluation.	Y. Wang, I.T. Phillips, R.M. Roberts, M. Haralick	Table Models	Pattern Recognition	37	2004	
7							
8							
9							
10							
11							
12							
13							

Fig. 2: Sample spreadsheet for references in journals

The following description of the spreadsheet-annotation is based on a simple usage-scenario which we introduce in this section. We assume that a special collection of references to journal articles has been created in form of a spreadsheet (illustrated in fig. 2). Those references might be collected by an expert of a particular domain.

The list is the result of a process involving a lot of personal expertise. In order to avoid redundant literature searches and to benefit from the expertise of the owner of the list (e.g. with respect to the filtering process and the quality of the references) other colleagues would like to share this resource as part of a “company knowledgebase”. In particular, they want to:

- § get access to the references collected by different colleagues based on the latest state of the list;
- § be sure about how to interpret this data (e.g. which kind of reference is given);
- § allow for machine-based interpretations of this data source (e.g. support for queries for particular references);
- § be able to query for references associated with a particular author.

With the help of semantic technologies, a common, machine-readable interpretation of the data could be established. To access the spreadsheets in terms of a resource for semantically interpreted data the following steps are necessary:

- § an ontology containing a suitable model for references to which those interested in the references can commit themselves to;

#### D2.4.1 Metadata Annotation Tools. Components version 1.

§ a possibility to create, store and maintain metadata which defines the interpretation of the spreadsheet in terms of the ontology;

§ means of access to the spreadsheet which always provide the actual data and includes additional data sets if the spreadsheet-table “grows”.

All those requirements can be met with the spreadsheet annotation tool. It allows selecting a spreadsheet and annotating it based on an ontology previously loaded. The annotation itself is range-based. The selected range must contain all relevant columns, which need to have (possibly implicit) headers. It can include rows which do not contain any data.

However, there are certain constraints/limitations for the annotation:

§ As explained in the previous section, the Spreadsheet-Plugin can currently handle only the simplest form of table-structure, which excludes merged cells and two-dimensional tables (as defined in REF) for example.

§ Once a range has been fixed, the data to be interpreted as instance data has to be within that range; i.e. if a table “grows” beyond this range, the new data will not be included;

§ There is no notification process for changes of the table-structure in terms of new columns which have been added.

§ As explained in the previous section, one spreadsheet table can only be interpreted in terms of a single concept. However, this limitation can be overcome with the help of the results in workpackage 4, namely the mapping tool. The same holds for the limitation that the annotation model does currently not support relations. If we consider for example to separate tables in a spreadsheet that have an implicit relation through one table column. The first table could contain a column “affiliation”, containing only a short name. The second table could list more data for each affiliation (name, location, ...). This implicit relation can be made explicit by the use of mediation technology (such as with OntoMap®).

The table shown in figure 2 contains data for journal references. For the sake of simplicity it does not contain all the data that is usually used for this kind of list (such as the numbers of the pages for example).

In the next sections we describe each of the steps necessary to annotate the table as well as a possibility to test the result of the process.

## D2.4.1 Metadata Annotation Tools. Components version 1.

### 2.1.4 Preparing the Annotation Process

The annotation process uses two views in the ontology editor: the navigator, representing the “ontology side” and the spreadsheet view, representing the “table side”. The ontology navigator is opened by default. The spreadsheet view can be opened via the top-level menu. Figure 3 shows how to open a view. The spreadsheet view can then be selected from a separate window as shown in figure 4.

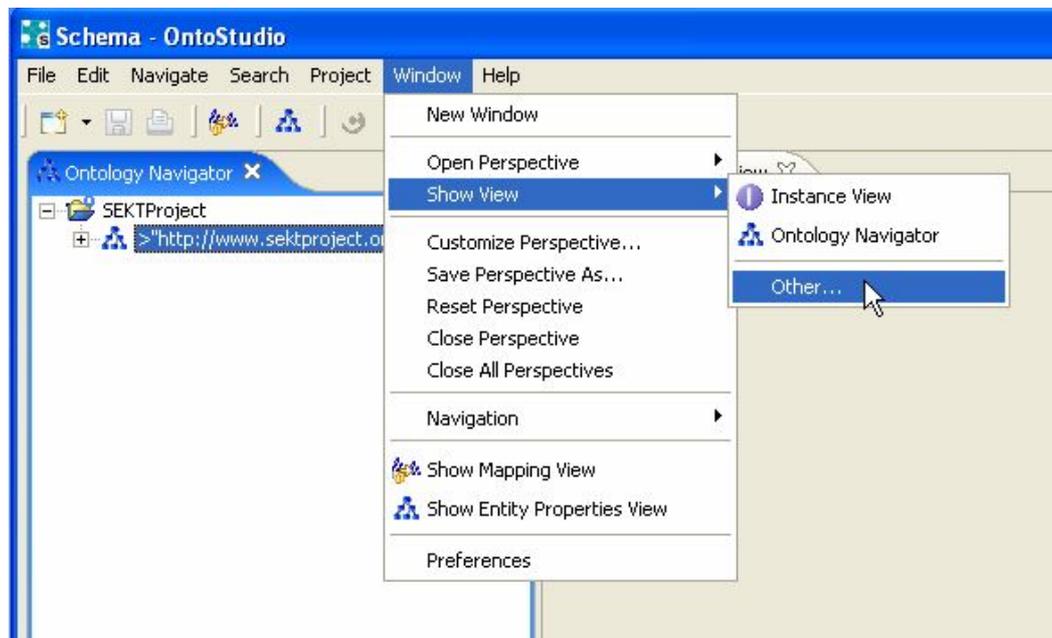


Fig. 3: Opening a view in OntoStudio®

#### D2.4.1 Metadata Annotation Tools. Components version 1.

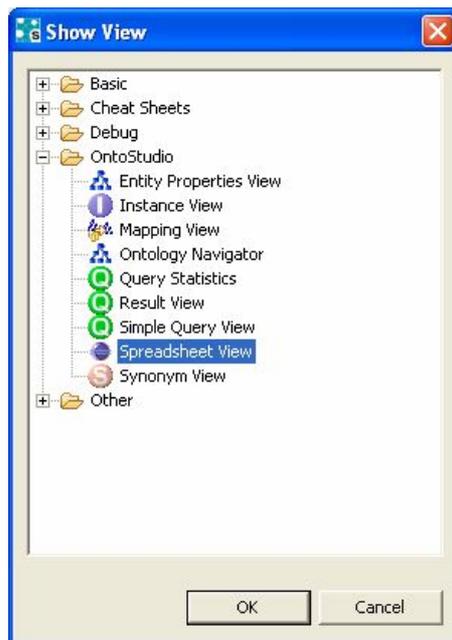


Fig. 4: **Selecting the Spreadsheet View**

The newly opened spreadsheet view contains an empty panel as well as a small menu with two buttons. We click on the right button (see figure 5) and select the spreadsheet file we want to annotate with the help of the file-chooser that appears. Then we open the concept tree in the navigator view. Figure 6 shows the fully configured workspace. We can now start with the actual annotation.

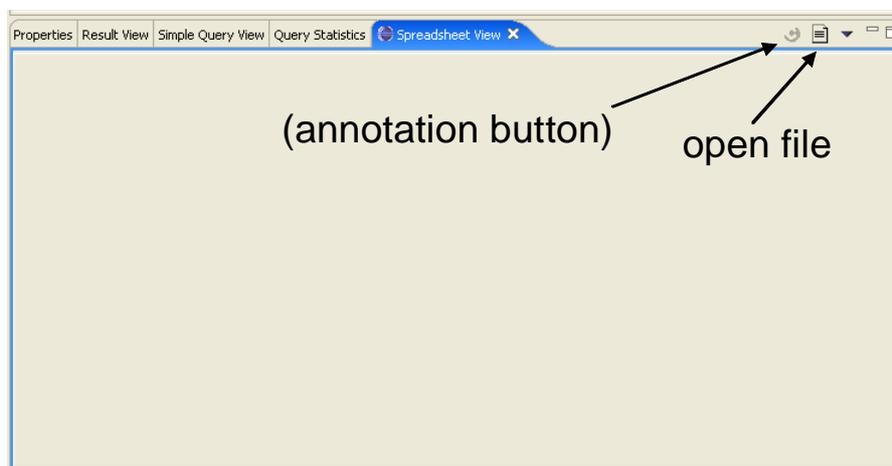


Fig. 5: **Empty Spreadsheet View**

## D2.4.1 Metadata Annotation Tools. Components version 1.

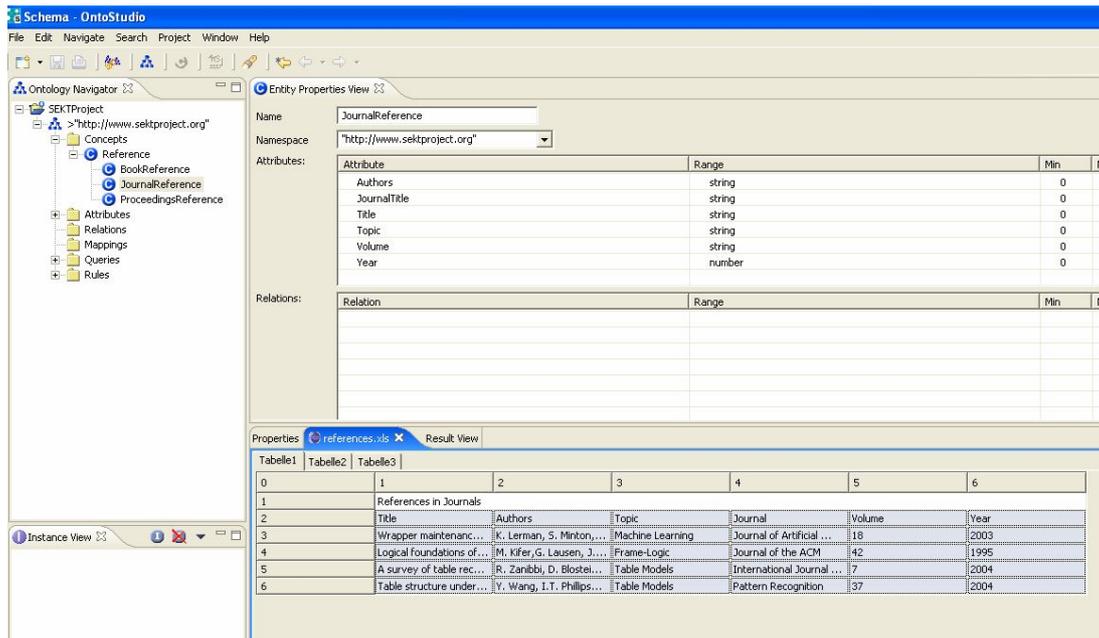


Fig. 6: Fully prepared workspace

### 2.1.5 Annotating an open spreadsheet

We select the concept that we want to associate with the spreadsheet-table. In our case this is the concept “JournalReference” (see figure 6). Afterwards we select the range we want to annotate with this concept. The selection of ranges works similar as in MS Excel® spreadsheets (selecting and dragging). Once the desired range has been marked we use the right mouse-button to get the annotation context menu (see figure 7).

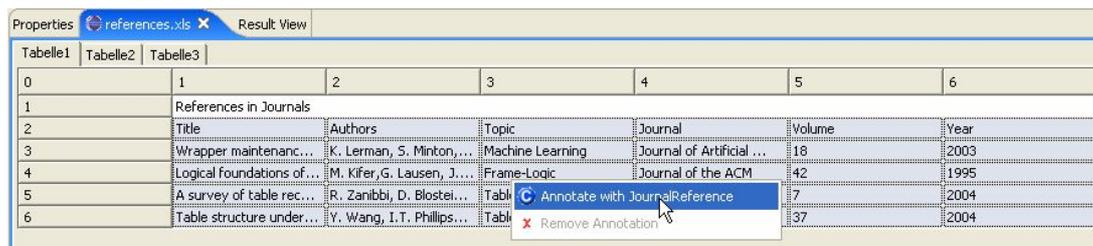


Fig. 7: Annotation context menu

If the context menu shows as first entry “no concept selected”, we would have to make sure that a concept is selected in the navigation tree. Otherwise, the selected concept appears in the context menu as show in figure 7. We now select the “Annotate...”-entry and open the annotation-window displayed in figure 8.

## D2.4.1 Metadata Annotation Tools. Components version 1.

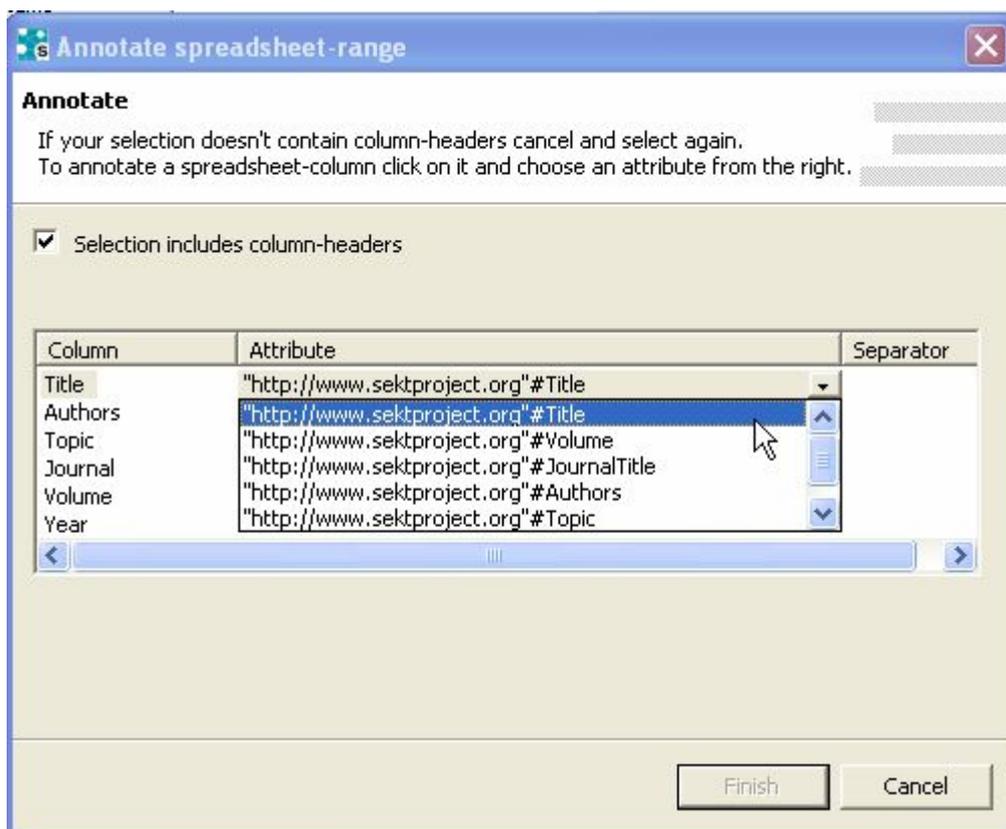


Fig. 8: **Attribute annotation in the annotation window**

The annotation window contains a table where the attributes of the selected concept can be mapped onto table-columns. To associate a spreadsheet-column with an attribute we first have to select the attribute in the most left column. Afterwards we can use the drop-down field in the “Attribute”-column to select the particular attribute. The drop-down is not available if the Column has not been selected properly.

If a spreadsheet-column contains values that are interpreted as multiple entries we have to specify a regular expression for separators. At this stage we ignore this possibility and map all columns onto attributes.

Since the range we selected includes the column-headers, we have to select the checkbox above the table. If there are no column-headers, the left column in the annotation window would display just the number of the column. Note that the annotation plugin does currently not support multiple headers<sup>4</sup>.

Figure 9 shows the result when all columns have been mapped onto an attribute.

---

<sup>4</sup> If we had for example one row of headers for the labels (e.g. “time”, “temperature”, ...) and another one with physical units (e.g. “sec”, “K”, ...) we would have to include just the lower row in the range we annotate.

## D2.4.1 Metadata Annotation Tools. Components version 1.

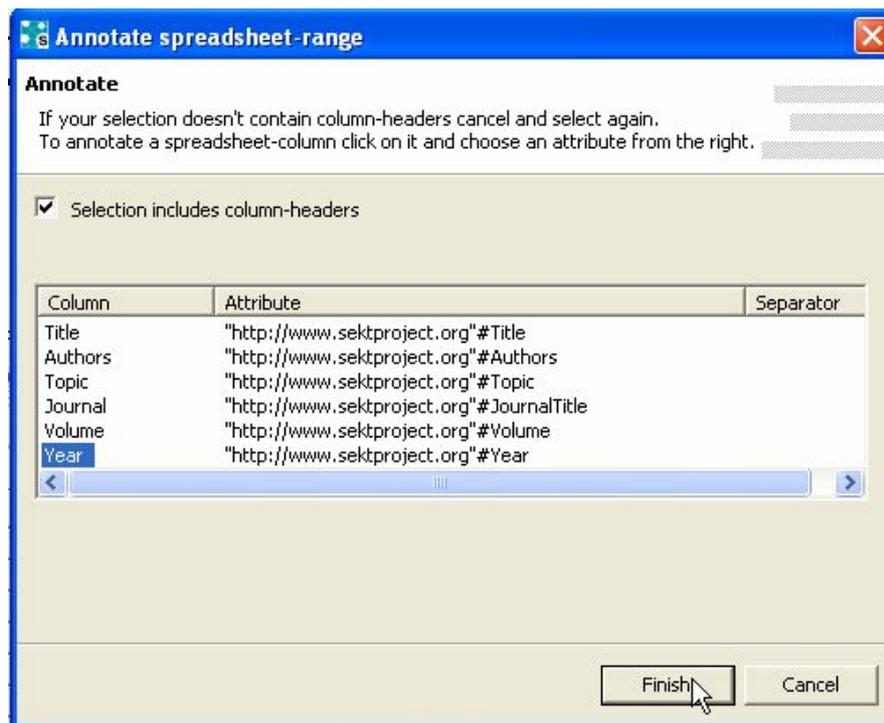


Fig. 9: Annotation of all columns

Once we select the “Finish”-Button, the annotation metadata will be generated in the background. This concerns basically the annotation rules that allow a “live” access and interpretation of the spreadsheet-table. If an OntoStudio-version is used that includes the rule-plugin, the generated annotation rules could be displayed. Those rules should not be changed manually to avoid inconsistencies in the annotation schema. Internally those rules are handled in the same way as other rules. The annotation-model used to generate and maintain the rules builds on the existing infrastructure at a higher level of abstraction. The annotation-model itself is reflected by the plugin user-interface.

### 2.1.6 Testing the Annotations

As previously described the annotation is based on a “live” interpretation of the spreadsheet table. The annotation meta-data indicates how to interpret the table (result of a design-time process) while the generation of instance data out of the actual table data is performed any time the instance data is requested (runtime process).

To test the annotations we have created we need to define a query against the ontology. OntoStudio supports the form-based creation of queries. Such a form-based query is always defined for concepts. A particular concept is the entry point for a query. With the help of the forms, conditions for attributes and relations can be defined. In our case, we need a simple query based on the “JournalReference”-concept we have annotated before.

## D2.4.1 Metadata Annotation Tools. Components version 1.

From the project-tree in the Navigator-View we select the “New Query” entry as illustrated in figure 10. OntoStudio® generates an empty query, which we rename to “JournalReferences”.

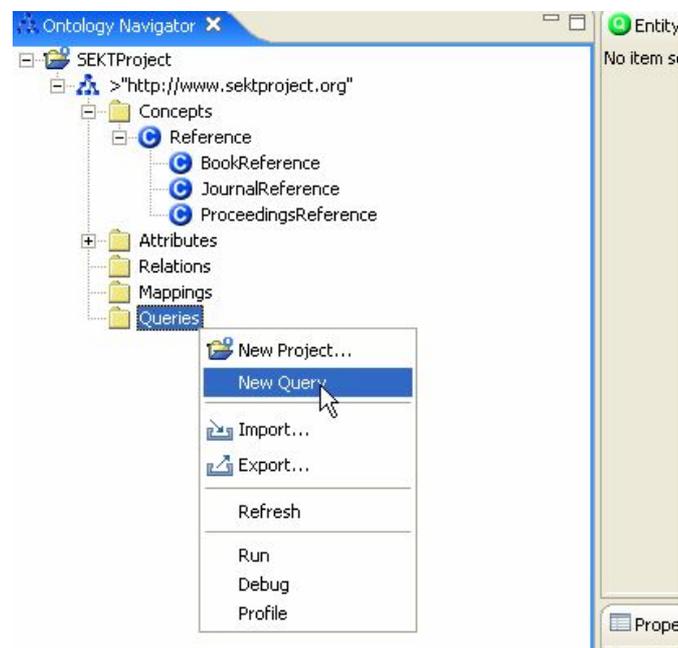


Fig. 10: Defining a new query

In the next step we select the new query in the Navigator-View and specify the concept whose instances we want to retrieve simply by double-clicking “JournalReference” in the list (see figure 11). We then define additional properties of the query using the Properties-View of the query. For each attribute of “JournalReference” we can specify if the attribute should be queried (and displayed) and if there is a restriction for the attribute. We use the check-box to include an attribute and ignore the possibility of defining restrictions. If no attribute has been selected, only the identifier of the concept will be retrieved and displayed<sup>5</sup>.

Finally we start the execution of the query as shown in figure 13. The result will be displayed in the Result-View (see figure 14) in the lower right of the workbench-window.

---

<sup>5</sup> Note that the selection of an attribute using the check-boxes not only determines what to display but also which instances to retrieve. If an existing instance of “JournalReference” has no value for “Authors” and we include “Authors” in the query, then this instance will not be part of the result due to the semantics of the query. This is a difference to relational database systems where one would get “NULL”-values. Future versions of OntoStudio will offer an option to use a similar mechanism.

## D2.4.1 Metadata Annotation Tools. Components version 1.

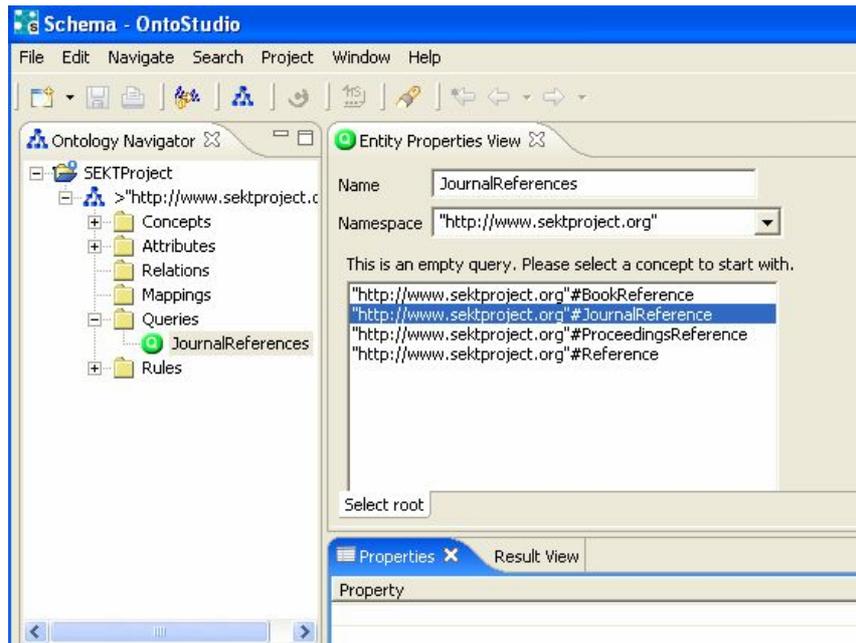


Fig. 11: Selecting “JournalReferences” for the query

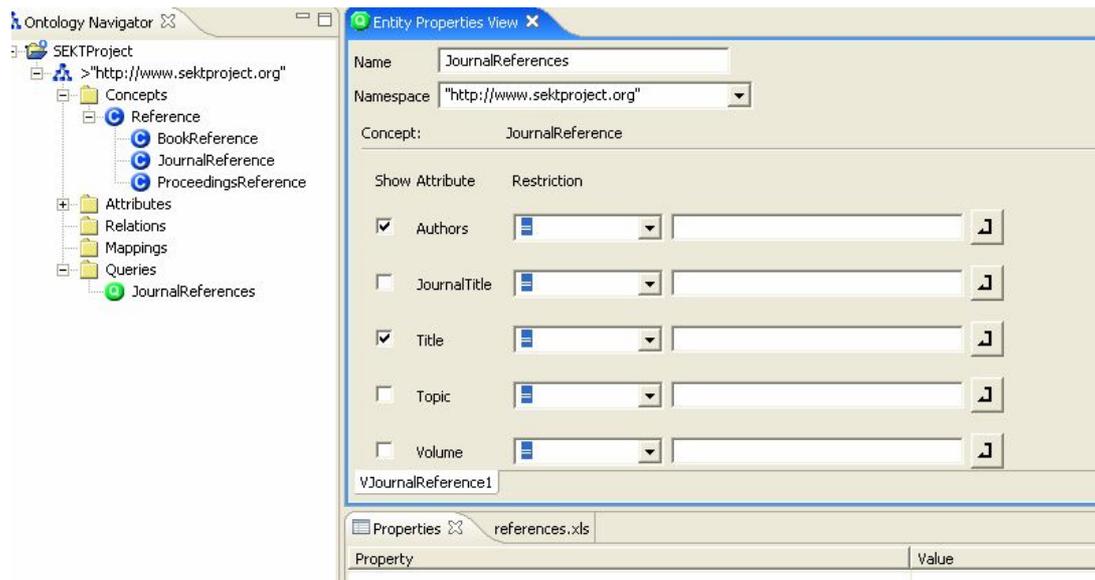


Fig. 12: Setting the query parameters

## D2.4.1 Metadata Annotation Tools. Components version 1.

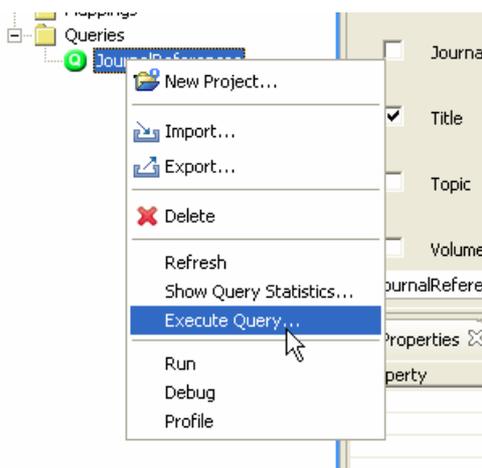
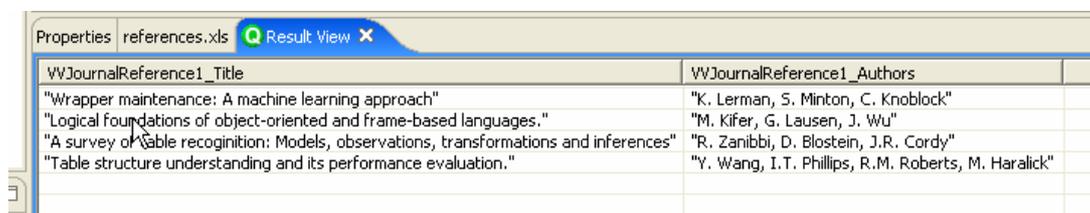


Fig. 13: Executing a query



WVJournalReference1_Title	WVJournalReference1_Authors
"Wrapper maintenance: A machine learning approach"	"K. Lerman, S. Minton, C. Knoblock"
"Logical foundations of object-oriented and frame-based languages."	"M. Kifer, G. Lausen, J. Wu"
"A survey of table recognition: Models, observations, transformations and inferences"	"R. Zanibbi, D. Blostein, J.R. Cordy"
"Table structure understanding and its performance evaluation."	"Y. Wang, I.T. Phillips, R.M. Roberts, M. Haralick"

Fig. 14: Query result (authors interpreted as single string)

In the result-list we can see that each entry in the “Authors”-column is interpreted as a single value for the “authors”-property of the “JournalReference”-concept. This does not fit with the semantics of the “authors”-property. Each author-name should be handled as a separate value. A query for authors should return a list of authors rather than a list of strings that is itself a list of authors (and has to be interpreted “outside” of the ontology).

The Spreadsheet-Plugin supports the usage of regular expressions for strings that have to be interpreted as a list of values. A regular expression defines what should be interpreted as a separator. In our case there is only a single comma that appears as separator. To use a comma as separator, we follow the procedure described in section 2.1.5 and insert a comma as regular expression for a separator of values (see figure 15). Afterwards we select the previously created query and execute it again. The result shows that the “authors”-string is now interpreted as a list of values (see figure 16).

The next section describes how we can slightly modify the setting. The workspace with the ontology should be left open for this modification.

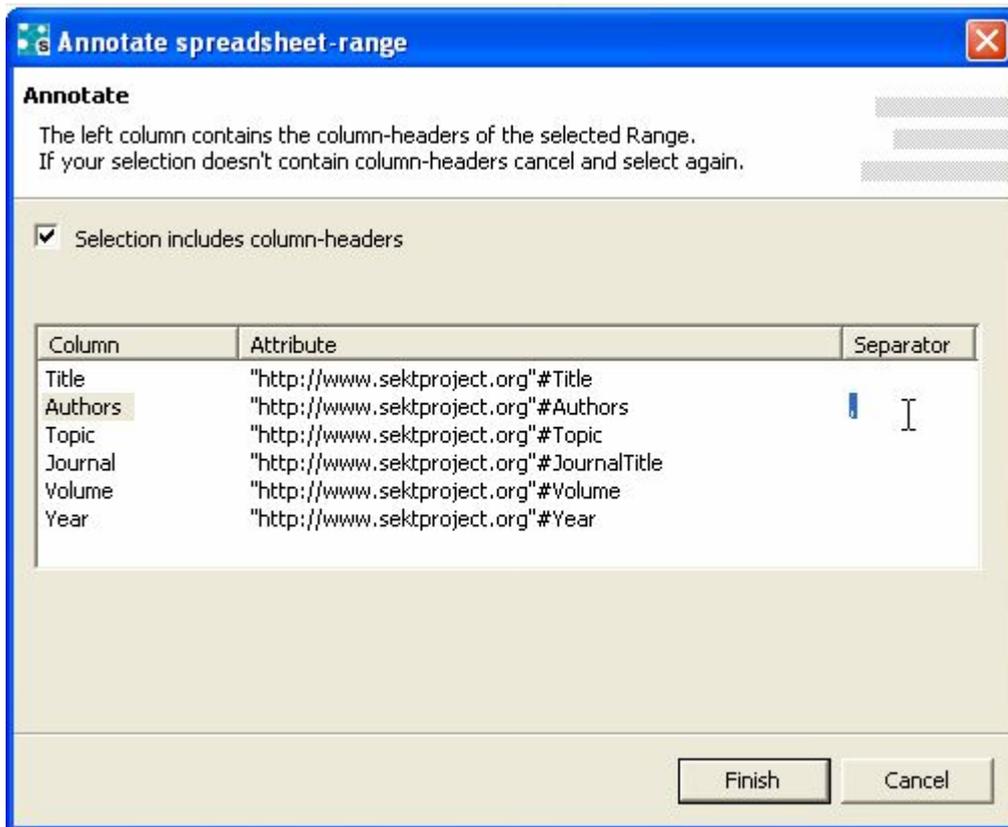


Fig. 15: Adding a separator expression to the “Authors”-annotation

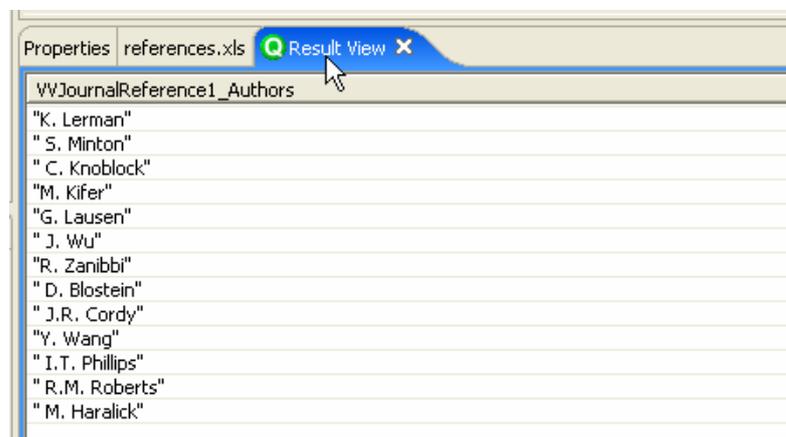


Fig. 16: Query result (authors interpreted as a list of values)

### 2.1.7 Modifications

In order to further investigate the possibilities of regular expressions and test the live-access. We open the spreadsheet and modify one ore more entries by changing the value in the “authors”-column. For the first entry, the value is replaced by the value “K. Lerman, S. Minton and C. Knoblock”. This makes it necessary to use a slightly more complex (but still simple) regular expression. Before we change the regular

expression, we save the Excel®-file and execute the “JournalReferences”-query again. In the result list we see an entry “S. Minton and C. Knoblock”. That means the cached data has been replaced by the actual spreadsheet data. In the next step we change the regular expression to “(,| and )”<sup>6</sup>. The result list after executing the query again should now contain the same results as in figure 16.

## 2.2 Text-based Annotations with OntoOffice® IE

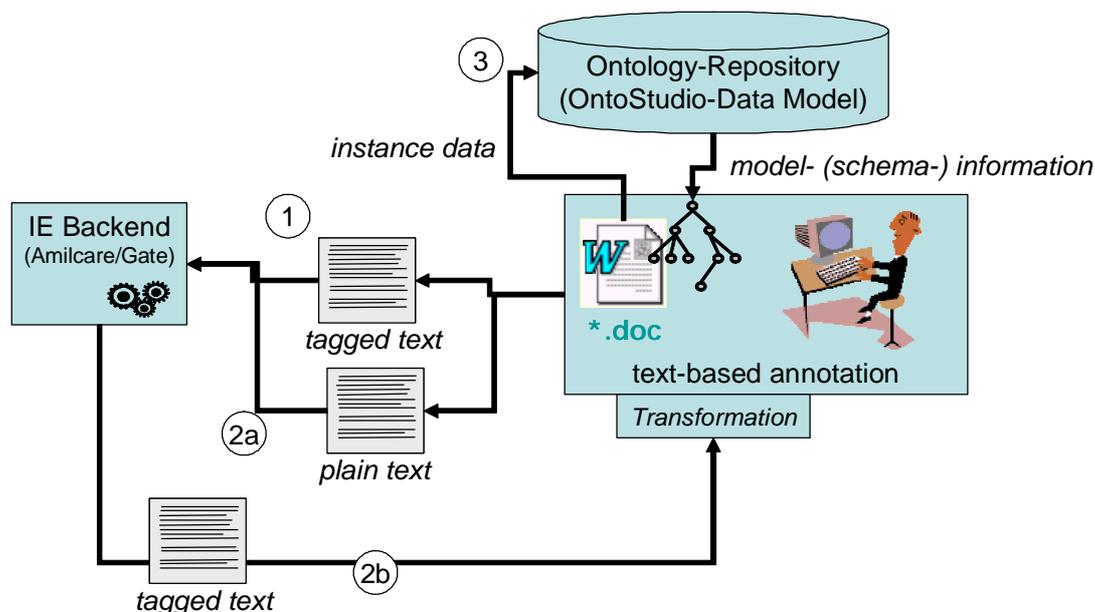


Fig. 17: Data flow for text-based annotations with OntoOffice® IE

The OntoOffice® IE plugin works in principle as a stand-alone tool for manual annotations, automatic annotations, the generation of training data for automatic annotations and instance data generation. The plugin itself does not implement the technology needed for the Information Extraction (automatic annotations, training), but it relies on the GATE-based Information Extraction -tool Amilcare (see [Cira03]). The latter is an implementation of mixed-initiative learning (see also [Li2004]).

Figure 17 illustrates the data flow for OntoOffice® IE:

- § (1) Manual annotations are transferred as training data for learning algorithms;
- § (2a) Existing documents are transferred for automatic annotations in form of plain text;
- § (2b) Automatically annotated data is transferred back to OntoOffice® (within the MS Word® environment);

<sup>6</sup> Note the two spaces around the word “and”; they are necessary to avoid matches in names containing “and”.

#### D2.4.1 Metadata Annotation Tools. Components version 1.

§ (3) Annotations are stored in the ontology engineering environment.

The basic idea of Information Extraction (IE) is to overcome the limitations of manual semantic enrichments of unstructured or insufficiently structured resources:

§ relatively high effort for the annotation process;

§ limited flexibility (adaption to frequent changes are expensive);

§ need for solid understanding of the annotation process (including the resources and the underlying model);

The IE system operating in the background of OntoOffice® IE is an adaptive IE system. It uses machine learning to adapt to new applications/domains, based learning algorithm inducing rules which extract information. Rules are learnt by generalising over a set of examples found in a training corpus annotated with XML tags. The system learns how to reproduce such annotation via Information Extraction.

##### *2.2.1 Annotation Model*

The current version of the plugin is based on an annotation model that associates a single instance of a particular concept with the content of the document. Consequently, the data for a single instance can be generated as a result of the annotation process. The background is that the IE-backend that is currently used expects the training data in such a form that a tagged document relates to an instance. However, this is likely to be changed in future versions of the plugin. Section 3 will briefly describe possibilities to extend the annotation model.

D2.4.1 Metadata Annotation Tools. Components version 1.

2.2.2 Scenario

Claims Report for Vehicle Claim v0.27

## Claims Report

13/05/2005

**Vehicle**

**Manufacturer:** \_\_\_\_\_

**Model year of vehicle:** \_\_\_\_\_

**Owner:** \_\_\_\_\_

**Accident/Incident**  
(Describe the accident; include date, place and weather conditions)

\_\_\_\_\_ I \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

**Damage to own Vehicle**  
(Describe the damage; include type and estimate of damage)

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

**Damage to other Vehicles**  
(Describe the damage; include type and estimate of damage)

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

**Comments**

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

Fig. 18: Claims Report

To illustrate the functionality of OntoOffice® IE, we use a scenario based a claims report for vehicle damages due to accidents. The description of relevant information has not been completely formalized (e.g. using electronic forms with drop-downs etc.) which gives people the freedom to use their own words and provide detailed

#### D2.4.1 Metadata Annotation Tools. Components version 1.

information if they want. However, it is assumed that certain facts are given, such as the place of the accident.

A further assumption is that a number of documents exists which have been filled in. Making the contents of the reports available in terms of semantically enriched data would result in several advantages:

- § report data could be queried e.g. for statistical analysis,
- § constraints on the data could be defined in a declarative way,
- § the formalizing process could be realized in an incremental fashion, keeping links to the original document.

To achieve this goal, we

- § annotate a number of documents manually,
- § generate training data for the automatic annotation,
- § use semi-automatic annotations for the rest of the document
- § store the annotations and the instance data.

The next sections describe how this can be achieved using OntoOffice® IE. We assume that a “mini ontology” modelling vehicle claims is available. We do not further describe this ontology since it is rather simple.

## D2.4.1 Metadata Annotation Tools. Components version 1.

### 2.2.3 *Preparing the Annotation Process*

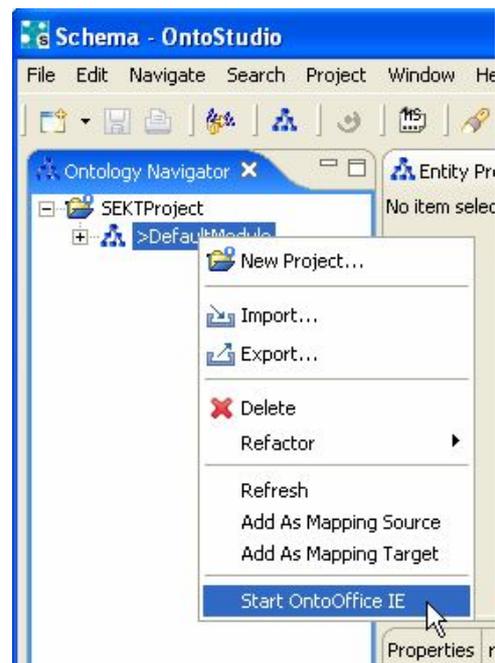


Fig. 19: **Starting OntoOffice® IE**

OntoOffice® IE is started from within OntoStudio® by using the right mouse-button on an ontology to open the context-menu and selecting the entry “Start OntoOffice IE” as illustrated in figure 19. Afterwards a file-chooser is opened to select the MS Word®-file we want to annotate. Figure 20 shows the loaded Plugin.

The plugin can also be started from within MS Word® and then be connected to an ontology server instance instead of the OntoStudio®-environment. However, we’ll not describe the procedure here.

The next step is the annotation of the document based on a concept of the ontology.

## D2.4.1 Metadata Annotation Tools. Components version 1.

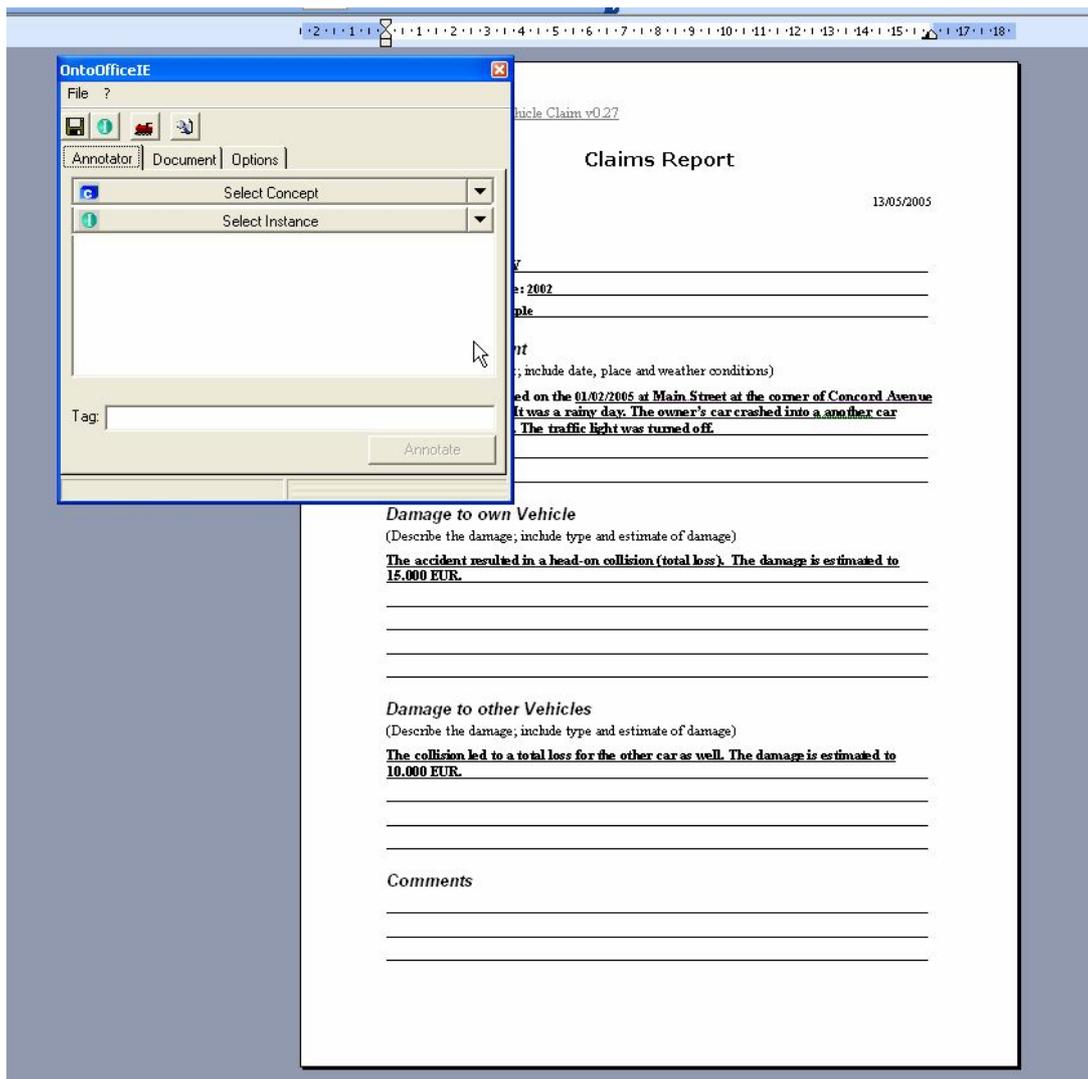


Fig. 20: Loaded OntoOffice® IE Plugin

### 2.2.4 Annotating a Document

Once the plugin is loaded, a concept has to be chosen using the “Select Concept” drop-down (see figure 20). In our case we want to associate an instance of a “VehicleDamageClaim” with the document. Once the concept is selected, its properties are shown (see figure 21). We can now select the part of the text that relates to a value for a certain property and then the property itself from the second drop-down. Figure 22 shows this for the “weather-conditions”. Finally we have to press the “Annotate”-Button.

OntoOffice® stores annotations in the document, using the SmartTag®-Technology of MS Word®. Existing annotations are indicated as SmartTags (a thin dashed line under the text and a SmartTag-“hover” as shown in figure 23). To save annotations, the document itself needs to be saved.

D2.4.1 Metadata Annotation Tools. Components version 1.

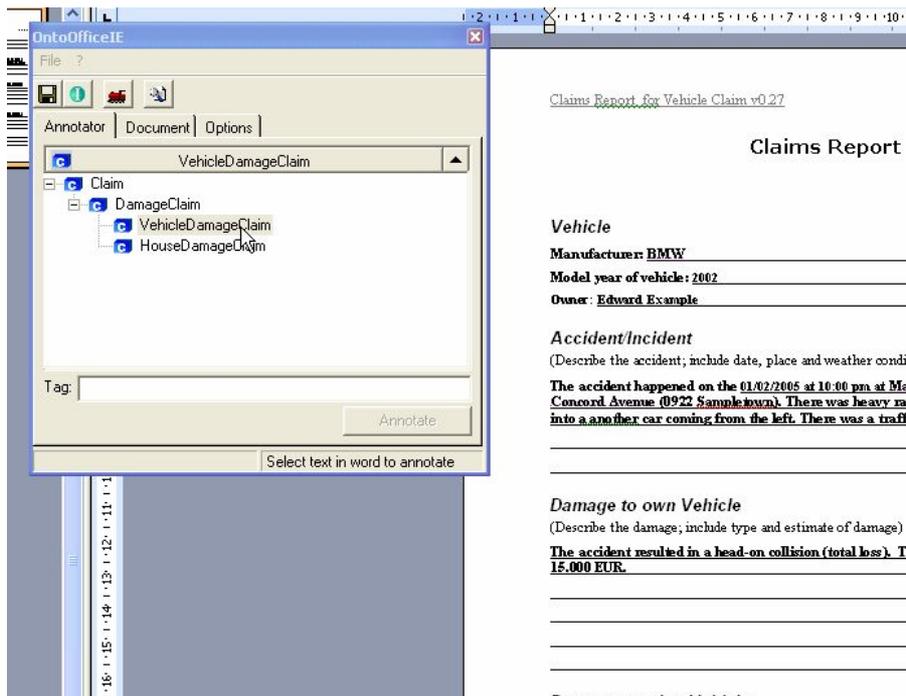


Fig. 21: Selecting a concept for the annotation

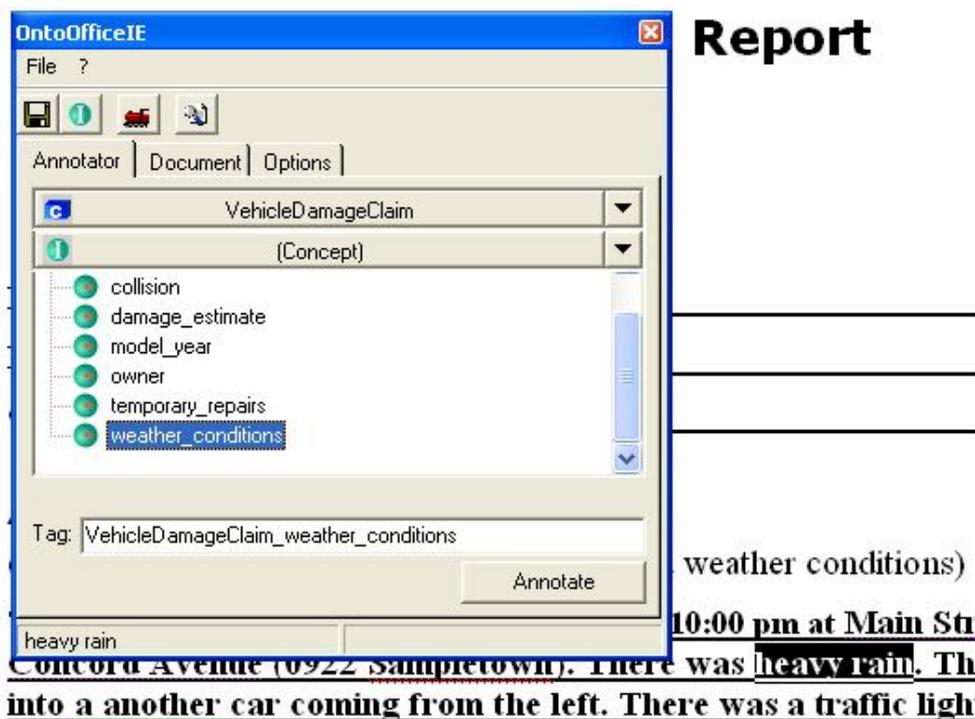


Fig. 22: Annotation of text

**005 at 10:00  at Main Street at th**  
**There was heavy rain. The owner**  
**ft. There was a traffic light which**

Fig. 23: Indication of an annotated text part

### 2.2.5 *Generating Training Data*

An annotated document can be used to provide training data for automatic annotations. The process is quite simple from a user's perspective.

Currently this works only for the document as a whole. To start the training process, we select the button . OntoOffice® IE generates a tagged text-document out of the contents of the Word-file and starts the training process of the IE-backend (Amilcare/GATE) using the settings of the working directory. The latter is a subfolder of the OntoOffice® delegator-plugin for OntoStudio (see section 2.2).

This starts the IE-backend in training mode in order to induce rules. The output of the training process is a set of rules which allows reproducing the annotation on texts of a similar type. It is stored in the working directory of the IE-backend.

### 2.2.6 *Automatic Annotation Process*

The automatic annotation of an open document can be started with the -button. OntoOffice® IE triggers the IE-backend with the extracted text of the document. The result is displayed in the document-tab of the plugin.

### 2.2.7 *Storing Instance Data*

In order to store the saved annotations as instance data, we have to press the -button. The result can be viewed using the Navigator View in OntoStudio®. We have to select the concept "VehicleDamageClaim" in the concept hierarchy and check the instance view in the lower left.

### **3 Discussion and Outlook**

The components introduced in this document allow users with little expertise in semantic technologies to enrich existing information resources with ontology-based metadata. They show how wide-spread office tools such as MS Word® and MS Excel® can be supported by semantic technologies in order to meet the requirements of common IT infrastructures. The interoperability with the OntoStudio ontology engineering workbench further increases the usability.

Within two simple scenarios it was shown how the components can be used in a process for the enrichment of existing sources. The case of the spreadsheet containing journal references is a typical example of a “personal knowledgebase” that is not available for other users in a form they would need it (supporting structured queries). This kind of knowledge base might contain valuable information for a large number of people, e.g. if the owner is an expert of a particular domain who filters references with significant relevance. The example shows how this kind of knowledge resource can be accessed based on a shared knowledge model.

An alternative to the annotation process within the second scenario might be the direct acquisition of semantic data, e.g. based on customized forms and controls. Supporting the use of natural language provides a higher degree of flexibility. The tools for annotation and information extraction offer a simple way to use the required technology.

The current versions of the two components have strong limitations. While this kind of end-user tool will always encapsulate great parts of the capabilities of the “backend technology”, it has to cover a large part of use cases occurring in typical organizations, probably leaving the more complicated cases to expert tools that provide the full functionality (e.g. command-line tools). For the spreadsheet annotation there is still room for extensions which make sense for non-expert users. This includes for example merged cells in form of recurring headers. The evaluation of possible patterns is currently in process. Version 2 of the spreadsheet annotation will support more complex annotation schemas. The support for the annotation of nested headers will always be limited, since the complexity of the annotation tool itself should not exceed a certain limit.

The text-annotation with OntoOffice® IE in its current version (and the data transfer as illustrated in figure 17) do currently not support multiple instances per document. To overcome this limitation, the support for section-wise annotations is currently evaluated. The latter might be especially useful for technical documents containing information about a number of entities of a similar type. The most important extension is the support for the OBIE IE system. As for the current IE system, the goal will be to keep as much configuration details hidden as possible and encapsulate background processes.

#### D2.4.1 Metadata Annotation Tools. Components version 1.

Finally there will always be functional limitations for the kind of end-user tools described in this document, due to the fact that complex tasks and concepts can be encapsulated only to a certain degree if the complexity should not be reflected by the user interface (which would contradict the motivation for those tools).

## 4 Bibliography and references

- [Cira03] CIRAVEGNA, F.; DINGLI, A.; WILKS, Y.; PETRELLI, D.: Using Adaptive Information Extraction for Effective Human-centred Document Annotation. Theoretical Aspects and Applications. In: FRANKE, J.; NAKHAEIZADEH, G.; RENZ, I. (Eds.): *Text Mining* New York, 2003, pp 153-165
- [Codd1970] CODD, E.A.: A relational model for large shared databanks. . *Communications of the ACM*, Vol. 13 (1970) No. 6, pp 377-387
- [Hurst2000] HURST, M: The Interpretation of Tables in Text. PhD thesis., University of Edinburgh, 2000
- [Kife1997] KIFER, M. AND LAUSEN, G.: FLogic: A higher-order language for reasoning about objects. *SIGMOD Record*, Vol. 18 (1997) No. 6, pp 134-146
- [Li2004] LI, Y.; BONTCHEVA, K.; DOWMAN, M.; ROBERTS, I.; CUNNINGHAM, H.: D2.1.1 Ontology Based Information Extraction (OBIE) v.1., SEKT deliverable, University of Sheffield 2004
- [Pivk2005] PIVK, A.; CIMIANO, P.; SURE, Y.: From Tables to Frames. . *Elsevier's Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, Vol. 3 (2005) No. 2-3, pp 132-146
- [Zani2004] ZANIBI, R.; BLOSTEIN, D.; CORDY, J.R.: A survey of table recognition: Models, observations, transformations and inferences. *International Journal of Document Analysis and Recognition.*, Vol. 7 (2004) No. (1), pp 1-16