An Active Annotation Support System for Regulatory Documents

Andreas Korger[1], Joachim Baumeister[1]

[1]University of Würzburg, Am Hubland, D-97074 Würzburg

Abstract

Manual document annotation is a resource intense task. The costs of annotation can be lowered by supporting the manual annotation with pre-processing of the available corpus and active in-process support of annotating users. To integrate different components into a coherent active annotation support system the XML Metadata Interchange standard can be used to exchange objects on the base of a meta-meta data model. Further, to integrate an existing knowledge graph into an annotation support system the RDF query language SPARQL can be used as an interface to analyze existent documents and declare new knowledge. In this manner the presented efforts contribute to structure and standardize the process of manual knowledge acquisition from regulatory documents.

Keywords

Knowledge Management, Document Annotation, Meta-Meta Data Models, SPARQL, Ontology Population, Natural Language Processing

1. Introduction

A core task in the field of knowledge management is to provide insight to documents related to the current problem situation. For example, for servicing industrial machines, the service technician will need quick access to the appropriate documentation. For internal knowledge management, large companies often provide access to regulatory compliance documents for use by their employees. This access is depicted conceptually on the left side of Figure 1. In these documents the textual parts are usually annotated by (semantic) metadata, in order to implement a quick and problem-oriented access to information. This semantic metadata in turn is then used by semantic search engines and navigation interfaces to provide a quick and context-oriented access [1]. Whereas for some new documents the authoring process can be extended by the augmentation of metadata, the metadata needs to be attached to legacy documents in any case. Albeit the progression in natural language processing, information extraction and ontology population, the attachment of metadata to document passages is often done manually in order to achieve high quality for the annotation. However, the manual annotation of documents is a cumbersome and costly task.

Available frameworks for general textual annotation lack active annotation support. In this work, we propose a semantic approach to actively annotate documents by integrating a domain specific ontology within the annotation task. The domain specific ontology represents prior

LWDA'22: Lernen Wissen Daten Analysen, 2022, Hildesheim

a.korger@informatik.uni-wuerzburg.de (A. Korger); joba@uni-wuerzburg.de (J. Baumeister)

CEUR Workshop Proceedings (CEUR-WS.org)

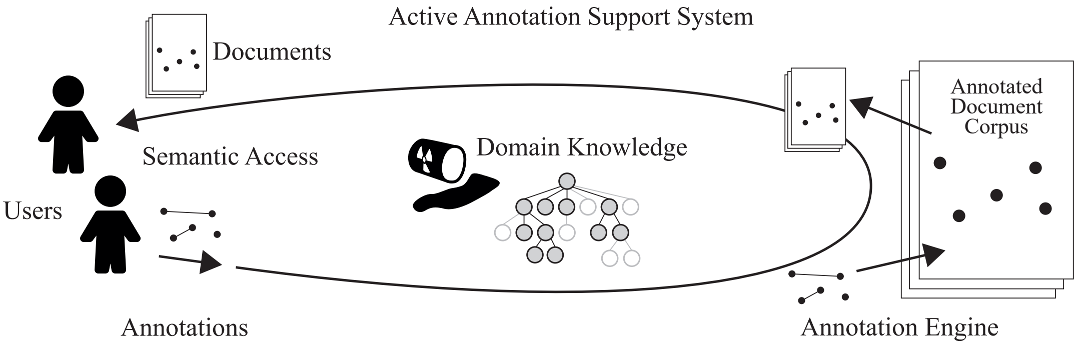

Figure 1: A conceptual view of annotating documents in knowledge management.

domain knowledge and it is used to integrate new knowledge collected in the active annotation process. The right side of Figure 1 shows a conceptual view of this approach. which reduces the efforts and improves quality of annotation compared to fully manual approaches.

The rest of the paper is organized as follows: In Section 2 we describe the components of the proposed framework as well as their interaction. In Section 3 we present a case study using the system for the annotation of regulatory documents for nuclear safety. We finish the paper with related work, future work, and a concluding statement.

2. Components of an Active Annotation Support System

In the following, we use an example with explanatory characteristics out of the domain of fire safety. A *fire* is an *incident* for which appropriate *measures* need to be taken. A *fire extinguisher* is such a measure but also a *blanket*. Such incidents and measures are in the scope of discovery for semantic annotation in unknown documents. An active annotation support system (AASS) suggests such annotations to the user with the recommendation of likely annotation features. It proposes automatically discovered annotations on the base of machine learning and natural language processing techniques, so-called *pre-annotations*. Such a system is called *active*, if the choice made by the user for the current annotation is incorporated into the next annotation recommendation. In this context, the system can also decide which textual passages are presented to the user for annotation to optimize the overall performance. The annotations are done on the basis of a semantic model for regulatory knowledge represented in an ontology. The semantic model is populated with instances created during the annotation step which makes up a knowledge graph of applied regulatory knowledge.

In this paper, an AASS for the knowledge management of regulatory documents is introduced, major components necessary for the implementation are pointed out and explained in detail. The workflow of bringing different components of the architecture together in a performant and consistent manner is explained. Basically there are two processing environments in use. First, the *annotation frontend* in which the user works and does his manual annotations. Second, the *backend* which does background work like file handling, provides NLP functionality, and manages consistency. A graphical view of the workflow and incorporated components is presented in Figure 2, components shown in the figure will be explained with more detail in the following sections.

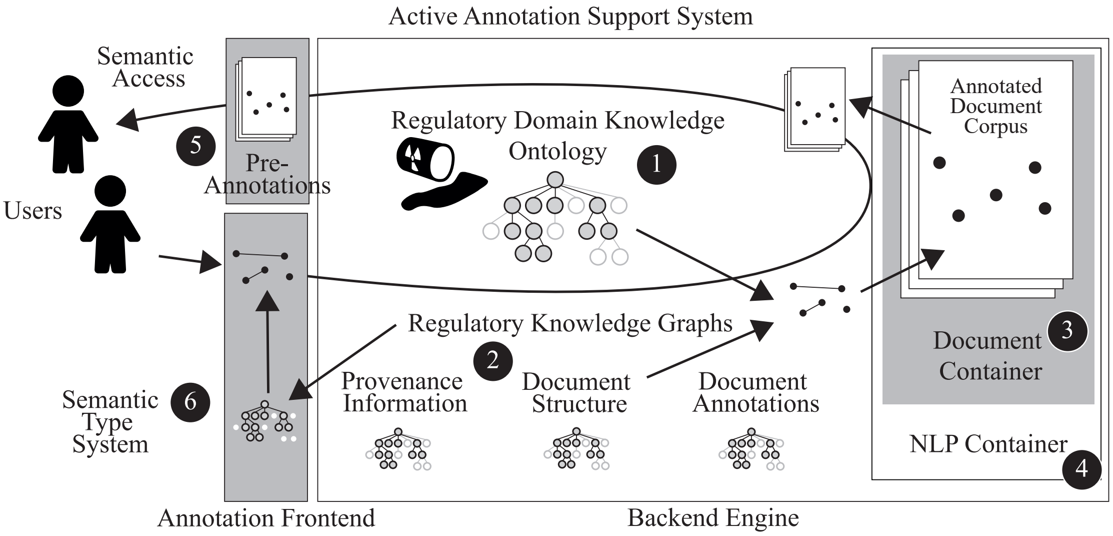

Figure 2: A view of the main components and their workflow for the active annotation of regulatory documents accessed by users over the frontend and processed in the backend.

2.1. Natural Language Processing Components

The handling of natural language text contained in regulatory documents requires appropriate techniques, namely to extract relevant entities and their relations [2]. For the processing of natural language text a *supporting NLP container* is necessary that provides functionality like text offset handling, tokenization, string matching, and rule matching. Each document is encapsulated in such a container on the backend side as Figure 2 shows in the component number (4). The container provides an interface to semantic knowledge contained in the regulatory domain knowledge ontology (1). It applies natural language processing steps to grant access to the document on a level of tokens and entities. Rule matching steps are applied to identify relevant known and unknown entities together with their relations.

The *document container* is created with the input of the document content (3). From the existent ontology structure (1) together with the already available annotations saved in the corresponding knowledge graph (2) a hierarchical type system is derived that is implemented into the annotation environment. Therefore, the information has to be transferred into a format that can be consumed and displayed by the frontend tools. This type system holds the supportive information for the user. In this way the options are represented that a user has for annotating entities and their relations. The functionality of the annotation environment is used, in order to retrieve feedback within the active learning step. As the presented architecture picks the best of diverse data models it is important to guarantee consistency of data. For the practical application the documents need to be uniquely addressable as well as different annotated versions of the same document need to be identifiable. Furthermore, the annotated metadata needs to be manageable. For instance, it is important to know which annotation was done by which annotating component. Therefore, a systematic model of unique identifiers is maintained. Additionally, meta information about the provenance of data is stored in the knowledge graph. The document container itself provides measures to maintain its own identifiers and assures at least the inherent (syntactical) correctness when transferring identifiers to the frontend.

Therefore, this container needs to communicate with the *annotation tool* for which a standard is necessary for the interchange of metadata. The pre-annotations (5) created by the NLP container are transferred to the document container which forwards them to the frontend. They are displayed for instance in the same manner as manual annotations are and can be accepted or adjusted by the user. The user might want to change the beginning and ending of the automatic annotation, the annotated features or delete the whole annotation. The annotations of the user are communicated to the backend side in the same manner as they are passed to the frontend.

2.2. Semantic Technology Components

Semantic components are modularized and stacked with the intention to re-use and adapt them to new regulatory domains. The fundament is an ontology (1) that represents the basic regulatory knowledge model. This model is instantiated and instances are aggregated in different knowledge graphs (2). Semantic technologies are used for three different purposes:

First, prior and learned domain knowledge is organized in the knowledge graph on the base of the knowledge organization model. For this task we use the SKOS (Simple Knowledge Organization System) [3] scheme and build a domain specific semantic model on top. The domain knowledge consists of entities that are described with labels and descriptions as well as proprietary relations between them. These are the entities and relations the user wants to discover and annotate in the available document corpus.

Second, a set of extraction patterns is matched to the corpus to identify unknown entities and relations. All these discovered information units are saved in the knowledge graph together with their annotations. An annotation made in a documents on the base of the domain knowledge signifies for instance, that a certain entity occurs at this position in the document. These automatic annotations are complemented by the manual annotations of the user. A big corpus quickly entails a large amount of annotations. This data has to be queryable to give insight with human perception capability for assessment of the quality of annotations and usage of the discovered knowledge. Additionally, a structured data storage in this manner allows for semantic search in the annotated data [2].

Third, provenance information is stored in the knowledge graph to organize for instance different authors, different documents, and experiments. We chose the Provenance Ontology (PROV-O) as a base to build a domain specific system above [4].

2.3. Textual Similarity Assessment

What distinguishes the annotation of regulatory documents from the annotation of general documents? We see the difference in the availability of a semantic model coding prior knowledge founding on several previous case studies. Further, this provides the base to construct a domain independent NLP engine exploiting textual phenomena that occur especially in texts of regulation. The combination of both facilitates the annotation of unknown regulatory texts. For instance, having taxonomies available of regulatory domain specific entities allows for recommending annotation features based on generalization and specification in the neighborhood defined by the topology of the taxonomy [5]. The domain specific relations and which textual indicators in regulatory speech point to them, simplifies the discovery of specific entities.

Being supported by regulatory semantic prior knowledge the "active" component of the annotation system can also exploit the users annotations efficiently. For instance, if the user has annotated a relation but not the entities the recommendation of fitting entities becomes possible due to domain knowledge. For instance, taken the example of fire safety "*... detecting and extinguishing quickly those fires which do start ...*". We already know, that a fire extinguisher is a measure against fires, this can be exploited by textual similarity assessment to identify that the verb extinguishing itself also is a measure. Additionally, *extinguishing* is connected with *detecting* via the conjunction *and* which allows the inference that detecting also is a measure and should be annotated as one. The adverb *quickly* can be added to create the more specific entities *detecting quickly* and *extinguishing quickly.* This would be stored in the knowledge graph as the measures *quick detection* and *quick extinguishment* in nominalized speech as well as the more general entities *detection* and *extinguishment.*

One additional aspect of having domain knowledge available is that specific similarity measures can be created. This is done either on the base of a language model like BERT or with the usage of a feature-based similarity measure [6]. In the case of the language model approach a pre-trained model is used. A model can also be trained on the specific textual data if sufficiently available. These similarity measures are then used to identify unknown textual passages that are similar to known textual passages that have already been classified. In the following section we describe an implementation of all components and report a case study in the domain of nuclear safety.

3. Case Study - Active Learning in the Domain of Nuclear Safety Regulations

Worldwide, several governmental and non governmental authorities provide regulatory compliance documents to assure nuclear safety. The scope reaches from recommendations for the safe operation of nuclear power plants to the safe execution of x-ray radiography. These documents are a valuable source of knowledge. To access this knowledge in a systematic manner, the availability of metadata within the documents is needed. This metadata has to be created by manual or automatic textual annotation. The information of how the regulatory metadata is structured is so-called *meta-metadata.* Synchronous with the annotation process, this knowledge about the structure of the metadata is improved. To evaluate and improve the before explained approach of active annotation support a case study in the domain of nuclear safety was put into practice. The framework was used to annotate selected documents of a corpus of 143 regulatory documents provided by the International Atomic Energy Association (IAEA) [7].

A task in this context is the annotation of potential incidents with the according safety measures as recommended in a document. For instance the phrase “*manual fire fighting*” is annotated as a measure to react to a discovered “*fire*”. Subsequently, it is wishful to provide this information in the next annotation step. This can be done by either annotating all occurrences of the phrase “*manual fire fighting*” or “on the fly” by providing the available information to the domain expert as a likely annotation to choose from. In this manner the knowledge is continuously improved with every working step of manual annotation. The overall view of tools and techniques used to implement the whole system as depicted in Figure 3.

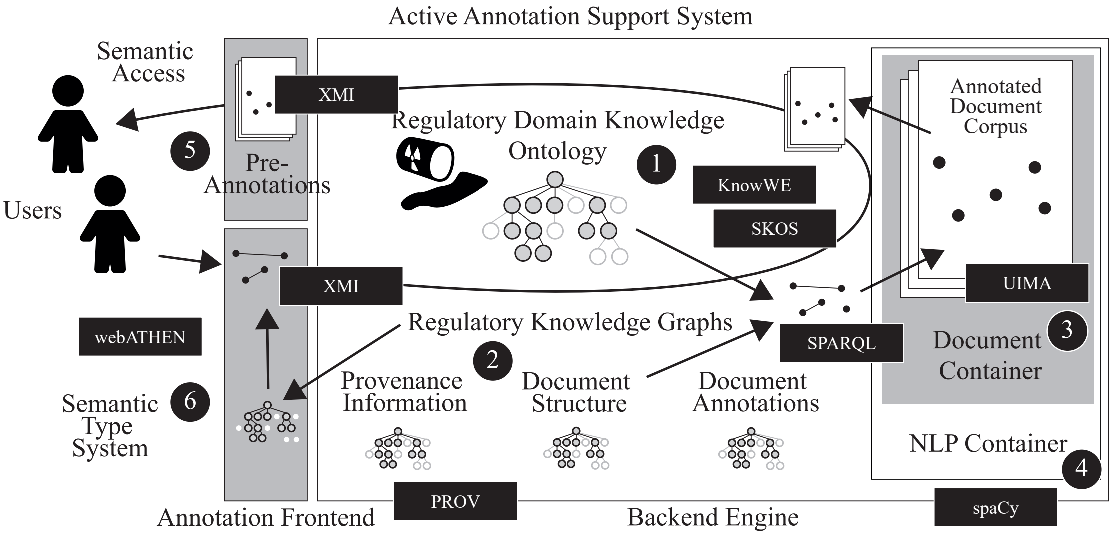

Figure 3: A view of used tools and techniques to implement an active annotation suport system for regulatory documents in the domain of nuclear safety.

3.1. The UIMA CAS Object and its Serialized Representation

Apache UIMA (Unstructured Information Management Architecture) is a framework for the management of unstructured information with the goal of structuring it into a processing pipeline of annotation steps [8]. In this setup UIMA is used as a document container in the backend depicted by component number three. UIMA provides functionality to hold the textual content of the document, the information about entities and their relations, and is capable of transferring this information into a serialized data format. For each document a so-called CAS object (Common Analysis Structure) is created with access to the type system schema necessary for the serialization and the communication to the frontend (5).

The CAS object maintains the correctness of indices and the according correctness of the serialized format. The XMI standard can be used for the serialized exchange of data objects on the base of meta-metadata models [9]. The content of an exemplary XMI file can be seen in the following Listing 1. The file holds the document content with its MIME type in the SOFA string (Subject Of Analysis) as well as two entities with the attributes *measureRoot* and *incidentRoot* and the relation between them with the label *isReactiveMeasureTo*.

```
<?xml version="1.0" encoding="UTF-8"?>
<xmi:XMI xmlns:xmi="http://www.omg.org/XMI" xmlns:cas="http:///uima/cas.ecore" xmi:version
    ="2.0">
   <ie:IEEntityGold xmi:id="2" sofa="1" begin="100" end="120" datatype="attribute" owlid="
    measureRoot" />
   <ie:IEEntityGold xmi:id="3" sofa="1" begin="180" end="190" datatype="attribute" owlid="
    incidentRoot" />
   <ie:IERelationGold xmi:id="4" sofa="1" begin="100" end="190" EntityFrom="2" EntityTo="3"
     Label="isReactiveMeasureTo" />
   <cas:Sofa xmi:id="1" sofaNum="1" mimeType="text" sofaString="Fire Safety in Nuclear
    Power Plants’’/>
   <cas:View sofa="1" members="1 2 3 4’’ />
</xmi:XMI>
```

Listing 1: A simplified XMI definition for a document of fire safety in nuclear power plants showing namespaces, entities, and relations.

The attribute names are extracted from the regulatory knowledge graph (2) and are proprietary to nuclear safety. Here, it is important to maintain consistency over the whole workflow to match UIMA, NLP, and knowledge graph entities. This task would exceed the capabilities of the document and NLP container functionality but can be conveniently fulfilled with the help of a knowledge management system like SKOS. To manually maintain the ontology and the belonging knowledge graphs an editor assists to assure correctness and supports manual ontology population. The tool (1) we chose for this task is KnowWE (Knowledge Wiki Environment) [10].

3.2. Natural Language Processing Components

A python library (pyCAS) [11] allows to integrate the document container build on the UIMA base with the NLP container (4). The main part of the NLP functionality is handled by the python-based NLP framework spaCy [12]. The system enables tokenization, part-of-speech-tagging, stemming, and rule-matching on the base of a pre-trained language model available in different languages. For similarity assessment of entities and textual passages the spaCy library is enriched by a pre-trained BERT model [6]. Similarity assessment is a core task of the workflow. Entity labels retrieved from the knowledge graph need to be matched against the natural document text. In this process basic natural text processing steps regarding for instance, spelling, word stems, and synonyms [2] is applied. Further characteristics of speech like the part-of-speech classification is exploited to discover sentences with potential relations. In Figure 4 the surface of the annotation tool is sketched. The available annotations are highlighted with gray color, pre-annotations with light gray color. Component (1) shows the type system proposing selection of available entities fitting to the textual passage “*fires which do start*” (3) in hierarchical order derived from the knowledge graph (2). How relations between entities are displayed is shown by component (4) which links two entities with the relation type *isReactiveMeasureTo*.

3.3. Querying Regulatory Knowledge

A benefit of the automatic and manual annotation efforts is that the annotated documents can be accessed via queries allowing, e.g., for semantic search on them. SPARQL is a query language

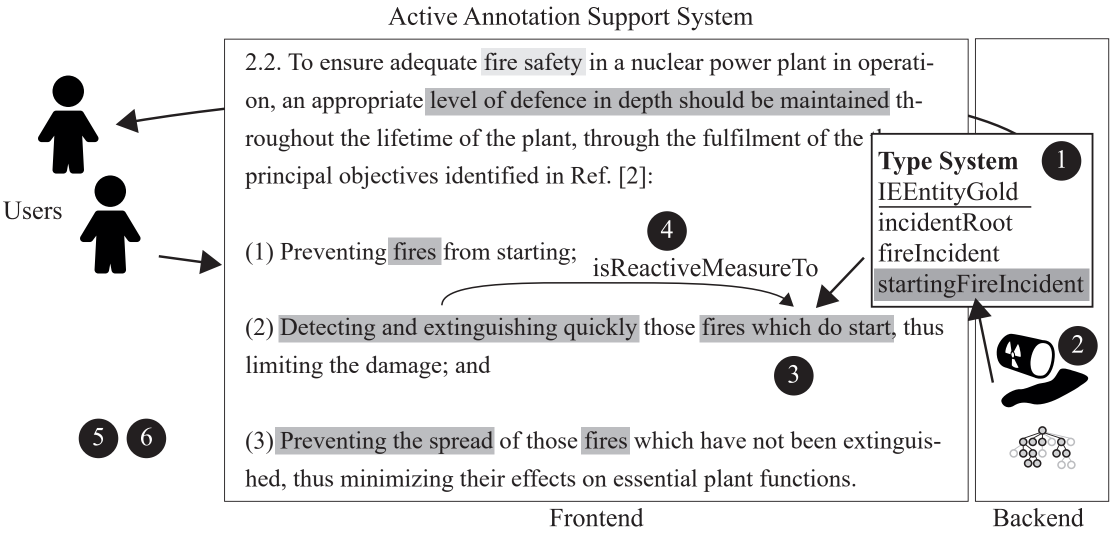
Active Annotation Support System
Users
2.2. To ensure adequate fire safety in a nuclear power plant in operation, an appropriate level of defence in depth should be maintained throughout the lifetime of the plant, through the fulfilment of the
principal objectives identified in Ref. [2]:
(1) Preventing fires from starting;
4
isReactiveMeasureTo
(2) Detecting and extinguishing quickly those fires which do start, thus limiting the damage; and
3
5
6
(3) Preventing the spread of those fires which have not been extinguished, thus minimizing their effects on essential plant functions.
Type System
1
IEEntityGold
incidentRoot
fireIncident
startingFireIncident
2
Frontend
Backend

Figure 4: Illustration of the graphical annotation surface for the work of manually annotating documents with features derived from the background knowledge and presented as a type system.

to retrieve and manipulate information contained in RDF knowledge graphs. Relevant features are extracted with a query and then stored in an appropriate data structure provided by the programming language (array, list, object, matrix, etc.). When the NLP processing is done, the results are transferred into the needed representation (CAS, XMI, RDF). In Figure 5 an excerpt of a taxonomy is illustrated. It orders incidents that are relevant in the domain of nuclear safety from a most general root incident *pirinu:incidentRootNuclearSafety* to more specific incidents connect via the property *piri:broader*. The property signifies that the entity in scope is of a more general character.

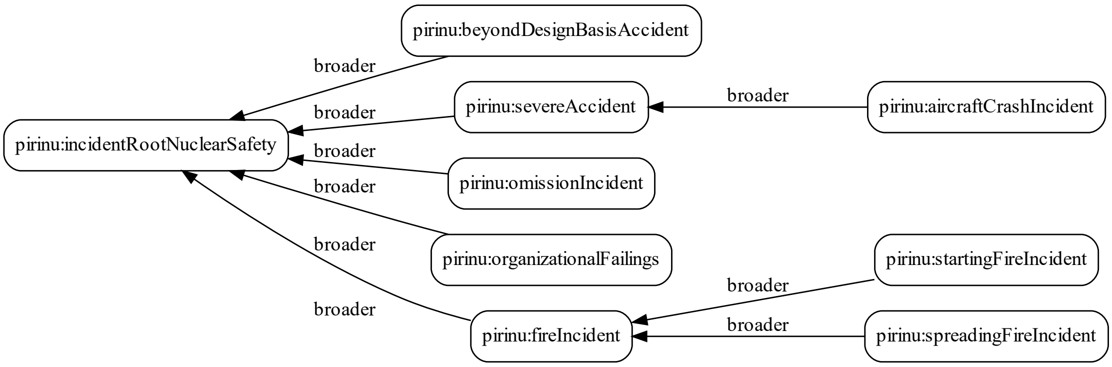
pirinu:beyondDesignBasisAccident
pirinu:severeAccident
pirinu:aircraftCrashIncident
pirinu:incidentRootNuclearSafety
pirinu:omissionIncident
pirinu:organizationalFailings
pirinu:startingFireIncident
pirinu:fireIncident
pirinu:spreadingFireIncident
broader

Figure 5: Excerpt of a taxonomy hierarchically structuring nuclear safety incidents.

The taxonomy shown before can be accessed via a SPARQL statement which is depicted in

Listing 2. The core ontology definitions are aggregated with the namespace *piri*. The knowledge graph for the domain of nuclear safety is separated with the proprietary namespace *pirinu* which stands for “piri nuclear”. The asterisk following *piri:broader* signifies that all entities that are transitively related to *pirinu:incidentRootNuclearSafety* with *piri:broader* should be retrieved.

```
SELECT ?x ?yLabel ?z
      WHERE {
      ?x ?y ?z .
      ?x piri:broader+ pirinu:incidentRootNuclearSafety .
      FILTER (?y = piri:broader).
      FILTER (?z != ?x).
      BIND (SUBSTR(STR(?y), 33) AS ?yLabel)
}
```

Listing 2: A SPARQL statement that selects all entities transitively related to pirinu:incidentRootNuclearSafety by the property piri:broader.

Most often a large number of annotations is created by an automatic annotation process. To manually browse and assess their quality, additional user support is necessary. The present corpus of nuclear safety consists of more than 10,000 pages of natural text. When fully annotated this quickly results in millions of annotations which exceeds human perception capabilities. Hence, the annotated knowledge needs to be aggregated and evaluated in a way complying to user requirements. Therefore, KnowWE is used for the presentation of the annotated data with a variety of options. Automatic annotations can be structured into a tabular format. The tabular structure can be adapted to the current context. Graphical data visualization is beneficial to display relations between annotated entities as depicted by Figure 5 .

Listing 3 shows an example of how to extract all annotations that are relevant for a specific user problem. The user wants to know about all entities and their relations that were manually annotated in a specific document for fire safety. The concept scheme *piri:IAEAGS* unites all annotations that where done manually and are approved by human review as gold standard annotations [13]. The query retrieves all information units with phrases, annotated features, text offsets, ids, and if available, relations having the current entity as an argument. The query is written and executed in the KnowWE environment on a wiki page. The result is visually presented with a tabular design and with additional filtering functionality. Users can comfortably browse manual and automatic annotations and edit specific entities in the annotation environment.

```
SELECT DISTINCT
?iu ?id ?offsetfrom ?offsetto ?phrase ?annotation
(GROUP_CONCAT(?relatedPhrase;SEPARATOR=",") AS ?relations)
WHERE {
  ?iu rdf:type piri:InformationUnit .
  ?iu piri:broader pirinu:regulatoryDocumentNuclearSafetyPub1091-web .
  ?iu piri:hasPhrase ?phrase .
  ?iu piri:hasOffsetFrom ?offsetfrom .
  ?iu piri:hasOffsetTo ?offsetto .
  ?iu piri:hasID ?id.
  ?iu piri:hasAnnotation ?annotation.
  ?iu piri:inScheme pirinu:annotationSchemeGoldAnnotationIAEA.
  OPTIONAL {
  ?iu piri:hasMeasure ?relatedMeasure .
  ?relatedMeasure piri:hasPhrase ?relatedPhrase .
    }
}
GROUP BY ?iu ?id ?phrase ?offsetfrom ?offsetto ?annotation
ORDER BY ?id
```

Listing 3: A SPARQL statement that selects all entities and their relations of the nuclear gold annotation scheme and presents them in an aggregated way.

4. Conclusions

This work presented the concept to actively support users in the task of manual document annotation. Namely, to improve performance and consistency of the annotating work. Necessary components were described in their interaction in a cyclic process. A focus was set on the modularity of the approach. This allows the exchange of individual modules to adapt the approach to other domains and new technologies. A case study in the domain of nuclear safety documents showed the practical application of the architecture. Specific tools in use where presented in their functionality. The use of the system in a life working setup on real world documents served as an evaluation revealing shortcomings and their refinement to the present status of quality.

4.1. Related Work

The XMI standard is used in a variety of scenarios. To use it for the modularization of an active annotation support system and the integration of a domain ontology of regulatory knowledge is a new approach. Software design shows parallels to the model-based annotation of documents as the reuse of existing components is saving resources in a same way programming code can be re-used and adapted to similar necessities. This aspect was elaborated with the use of the XMI standard by Di Felice et al. [14]. Interesting work on how to consistently explore XMI files for the generation of test cases was presented by Achimugu et al. [15]. Bucko et al. [16] presented a work towards the automation of model driven system architecture to support system architects in their work by an ontology model coding manual transformation guidelines. Nasiri et al. [17] try a similar approach to the present one on user stories for software modeling and extract class diagrams into a XMI file with natural language processing steps. Wardhana et al. [18] use the XMI standard as a bridge to transform a System Modeling Language diagram automatically into an ontology to replace to costly and error-prone manual transformation by system engineers. A work to transform business process models coding domain knowledge via decision learning support and the XMI standard into an UML model to assist software engineering was presented by Mythily et al. [19]. Previous work has also been done by the authors that gives more insight into the domain of nuclear safety and the according knowledge management [20]. Additional information together with the components of the architecture can be found on the PIRI website [21] and on github [22].

4.2. Future Work

Currently the integration of knowledge graphs into the UIMA proprietary type system lacks automation and standardization. This feature would make the overall architecture more seamless and also improve consistency. Furthermore, it would allow to reduce the efforts to, for instance, integrate any SKOS-based knowledge graph. Besides that, an extension of the CAS object to hold partial knowledge extracted by a SPARQL query in a standardized way would facilitate diverse processing steps.

References

[1] R. Guha, R. McCool, E. Miller, Semantic search, in: Twelfth International World Wide Web Conference (WWW 2003), 2003.

[2] D. Jurafsky, J. H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 1st ed., Prentice Hall PTR, USA, 2000.

[3] W3C, SKOS Simple Knowledge Organization System Reference: http://www.w3.org/TR/skos-reference, 2009.

[4] W3C, PROV-O: The PROV Ontology: http://www.w3.org/TR/prov-o, 2013.

[5] R. Bergmann, Experience Management, Springer, Berlin, Heidelberg, 2002.

[6] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics, Stroudsburg, PA, USA, 2019, pp. 4171–4186. arXiv:1810.04805v1.

[7] International Atomic Energy Agency, 2022. URL: https://www.iaea.org.

[8] D. Ferrucci, A. Lally, Accelerating corporate research in the development, application and deployment of human language technologies, in: SEALTS '03: Proceedings of the HLT-NAACL 2003 workshop on Software engineering and architecture of language technology systems, Association for Computational Linguistics, Morristown, NJ, USA, 2003, pp. 67–74.

[9] M. Weiss, XML Metadata Interchange, Springer US, Boston, MA, 2009, pp. 3597–3597.

[10] J. Baumeister, J. Reutelshoefer, F. Puppe, KnowWE: A semantic wiki for knowledge engineering, Applied Intelligence 35 (2011) 323–344.

[11] A. Zehe, Python implementation of the apache UIMA CAS data structure (2020). URL: https://gitlab2.informatik.uni-wuerzburg.de/alz20ij/PyUIMA.

[12] M. Honnibal, I. Montani, S. Van Landeghem, A. Boyd, spaCy: Industrial-strength Natural Language Processing in Python, Zenodo (2020).

[13] L. Wißler, M. Almashraee, D. Monett, A. Paschke, The gold standard in corpus annotation, in: IEEE GSC Passau, 2014.

[14] P. Di Felice, G. Paolone, R. Paesani, M. Marinelli, Design and Implementation of a Metadata Repository about UML Class Diagrams. A Software Tool Supporting the Automatic Feeding of the Repository, Electronics 11 (2022).

[15] O. Achimugu, P. Achimugu, C. Nwufoh, S. Husssein, R. Kolapo, T. Olufemi, An improved approach for generating test cases during model-based testing using tree traversal algorithm, Journal of Software Engineering and Applications 14 (2021) 257–265.

[16] B. Bučko, K. Zábovská, M. Zábovský, Ontology as a modeling tool within model driven architecture abstraction, in: 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2019, pp. 1525–1530.

[17] S. Nasiri, Y. Rhazali, M. Lahmer, A. Adadi, From user stories to UML diagrams driven by ontological and production model, International Journal of Advanced Computer Science and Applications 12 (2021).

[18] H. Wardhana, A. Ashari, A. Sari, Transformation of SysML requirement diagram into OWL ontologies, International Journal of Advanced Computer Science and Applications 11 (2020).

[19] M. Mythily, S. Saha, S. Selvam, I. T. J. Swamidason, BPM supported model generation by contemplating key elements of information security, Automated Software Engineering 29 (2022).

[20] A. Korger, J. Baumeister, Case-based generation of regulatory documents and their semantic relatedness, in: K. Arei, S. Kapoor, R. Bhatia (Eds.), Future of Information and Communication Conference San Francisco, volume 1130 of *Advances in Information and Communication*, Springer, 2020, pp. 91–110.

[21] A. Korger, J. Baumeister, Piri ontology, 2022. URL: https://www.piri-safety.com.

[22] A. Korger, J. Baumeister, 2022. URL: https://github.com/regdoc/piri.