Intelligent Information Management Targeted Competition Framework
ICT-2011.4.4(d)

Project	**FP7-318652 / BioASQ**
Deliverable	**D3.3**
Distribution	**Public**

http://www.bioasq.org

Annotation Tool

Axel-Cyrille Ngonga Ngomo, Norman Heino, René Speck, Timofey Ermilov, George Tsatsaronis

Status: Final (Version 1.0)

February 2013

Project

Project ref.no.	FP7-318652
Project acronym	BioASQ
Project full title	A challenge on large-scale biomedical semantic indexing and question answering
Porject site	http://www.bioasq.org
Project start	October 2012
Project duration	2 years
EC Project Officer	Martina Eydner

Deliverable

Deliverabe type	Report
Distribution level	Public
Deliverable Number	D3.3
Deliverable title	Annotation Tool
Contractual date of delivery	M4 (January 2013)
Actual date of delivery	February 2013
Relevant Task(s)	WP3/Task 3.3
Partner Responsible	ULEI
Other contributors	TI
Number of pages	13
Author(s)	Axel-Cyrille Ngonga Ngomo, Norman Heino, René Speck, Timofey Ermilov, George Tsatsaronis
Internal Reviewers	Sergios Petridis, Makis Malakasiotis
Status & version	Final
Keywords	Annotation tool, WP3, RDF verbalization, Semantic search

Contents

List of Figures

List of Tables

1 Introduction

The annotation tool was created with the aim of supporting the creation of the benchmark data for the challenge Tasks 1b and 2b. As specified in the Description of Work presented to the unit, this tool was thus specifically designed to enable the biomedical experts to create the gold standards for these tasks. In its current version 1, the tool enables its users to:

- create evaluation questions or continue working on existing questions,
- search for relevant concepts, documents and triples that allow answering the questions,
- associate the evaluation questions with gold standard answers.
- annotate them with concepts from designated taxonomies or ontologies, and
- associate the answers with relevant triples and snippets from selected data sources.

The BioASQ annotation tool is based on existing semantic search technologies developed by TI and an RDF verbalization framework developed by ULEI. Moreover, it relies on the datasets presented in deliverable D3.2. After a brief comparison of our annotation with some state-of-the-art annotation tools, we give an overview of the tool itself. We focus especially on presenting its architecture and the services upon which it relies. The tool is available at `http://at.bioasq.org`. The sources for the tool presented herein as described by Heino et al. (2013). The SPARQL2NL framework is documented by **?**Ngonga Ngomo et al. (2013).

2 Related Work

Several tools have already been created with the goal of enabling the annotation of data. We evaluated a number of these tools with respect to the requirements based on the specific goals of the tool presented in the introduction. Especially, we required the tools to:

- be *open-source* to ensure that they could be extended with the required functionality at will,
- be *Web-based* due to the distribution of the annotators, i.e., our bio-medical experts,
- support the *export of the annotations as RDF*, as such an export is the basis for the later integration with the social network
- support the *manual creation of new items to annotate* as required for the creation of questions for the BioASQ tasks,
- allow the *annotation with concepts, RDF triples and snippets* as required for the creation of gold answers for the BioASQ tasks
- support the *integration of JSON search services* as required for the search through the data corpus from which the answers are to be retrieved by the competitors.

The results of our evaluation of the tools and frameworks Gate[1], Stanford Manual Annotation Tool[2], AKTive Media[3], WordFreak[4], Vogon[5], MMAX2[6] and UAM Corpus Tool [7] are shown in Table 2.1. Our evaluation showed that none of the existing tools was fit to be used for our purposes. Especially, most of them lacked RDF support. Moreover, the concept behind the annotation tool needed for BioASQ differs from the classical approach to annotation: Most of the tools supported the annotation of a predefined

[1] http://gate.ac.uk/
[2] http://nlp.stanford.edu/software/stanford-manual-annotation-tool-2004-05-16.tar.gz
[3] http://eprints.aktors.org/537/01/poster-camera.pdf
[4] http://wordfreak.sourceforge.net/
[5] http://sourceforge.net/projects/gobtan/
[6] http://mmax2.sourceforge.net/mmaxpaper.pdf
[7] http://www.wagsoft.com/CorpusTool/

	T1	T2	T3	T4	T5	T6	T7	T8	T9
Open Source	+	+	+	+	*O*	+	+	+	−
Web-Based	−	−	−	−	*O*	−	+	−	−
RDF export	+	−	−	−	+	−	+	−	−
Manual creation of corpus	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*	*O*
Annotation with concepts	+	+	+	+	+	+	+	+	+
Annotation with snippets	+	−	+	−	−	−	−	−	−
Annotation with triples	−	−	−	−	−	−	−	−	−
Search	*O*	−	*O*	−	−	−	−	*O*	*O*

Table 2.1: Evaluation of existing tools. T1 = Gate, T2 = Stanford Manual Annotation Tool, T3 = AKTiveMedia, T4 = WordFreak, T5 = OntoMat, T6 = Coat, T7 = Vogon, T8 = MMAX2, T9 = UIMA Corpus Tool. $+$ stands for "requirement fulfilled", O for requirement partly fulfilled and $-$ for requirement not fulfilled.

corpus. Thus, they do not provide interfaces for the creation of the corpus at runtime. Rather, they allow to load an existing corpus (that is to be annotated) prior to the annotation process itself. Yet, in the case of BioASQ, our goal is to *create a corpus of questions to annotate* and to *annotate the questions with snippets, documents, triples and concepts* concurrently. We thus decided to implement the annotation tool by relying on existing JavaScript libraries.

3 Annotation Tool

In this section, we present the BioASQ annotation tool in more detail. We begin by giving an overview of its architecture. Then, we present the functionality of the tool by presenting each step of the annotation process. Finally, we give some insights in the services that the tool uses in the background.

3.1 Overview

The developed tool consists of four main components as depicted in Figure 3.1.

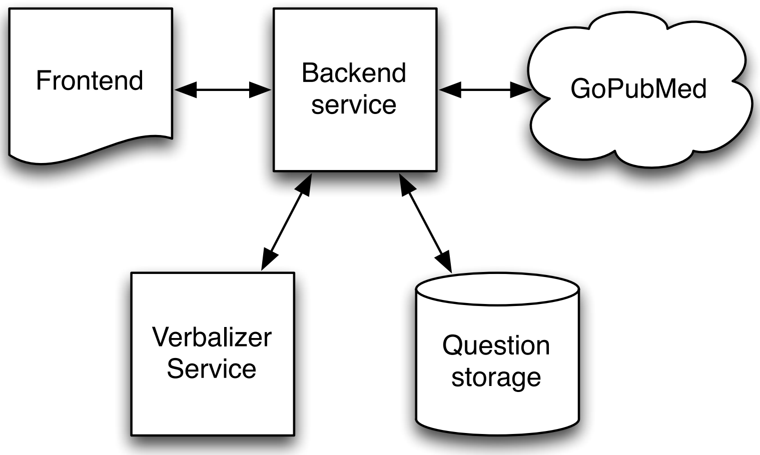


Figure 3.1: Architectural overview of the developed annotation tool.

The main annotation workflow is driven by the *frontend* component that resides in the user's browser. It consists of HTML pages along with some client-side scripts. Functionality that requires persistent storage (authentication) or interfacing with remote hosts (e.g., search) is supported by the *backend ser-*

vices. These are implemented as lightweight Node[1] services relying entirely on the REST paradigm with JSON being the only message format. All persistent storage is achieved using MongoDB[2], a high-performance NoSQL database. The JSON (or more precisely BSON) documents stored in MongoDB resemble RDF/JSON documents in the developed vocabulary, thus alleviating RDF export. All search requests are sent to individual GoPubMed services. The results are aggregated and, in the case of RDF statements, verbalized as described in section 3.4.

3.2 Functionality

The BioASQ annotation tool was implemented so as to be aligned with the annotation guidelines that were decided upon with the biomedical expert annotators. Accordingly, it supports the following five basic steps: authenticate, search, select, annotate and store. In the following, we present each of these steps.

3.2.1 Authentication

The basic requirement behind the authentication process was to allow only the experts from the biomedical expert team to create the benchmark. Thus, the backend of the annotation tool manages a whitelist containing emails of users who are allowed to use the tool. The whitelist was conceived to be easily extensible by the means of simple JavaScript commands [3] A user that aims to register has to provide the tool with a name, email and password through the frontend (see Figure 3.2). If he/she is in the whitelist, then he/she receives a welcome email with an activation link. Once this link was used in a browser, the account is activated and the browser redirects the user to the login page. The login is then carried out by using his email and password (see Figure 3.3).

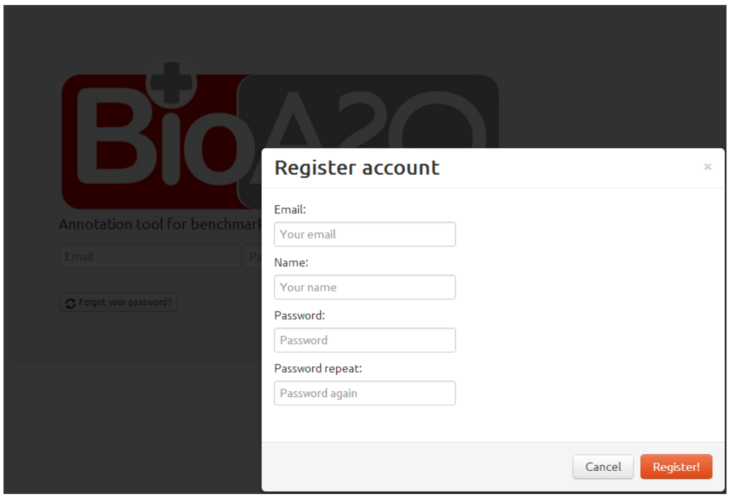


Figure 3.2: Screenshot of the annotation tool's registration screen.

[1] http://nodejs.org
[2] http://www.mongodb.org
[3] See https://github.com/AKSW/BioASQ-AT for all details.

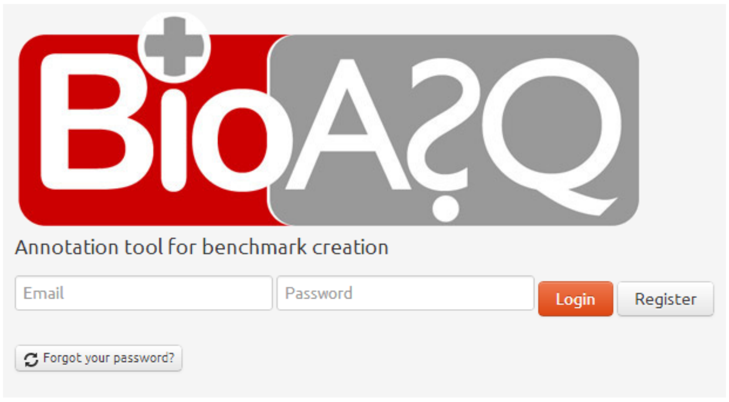


Figure 3.3: Screenshot of the annotation tool's login screen.

3.2.2 Search

The search interface as seen in Figure 3.4 accepts a number of keywords that are sent in parallel to each of the the GoPubMed services. Upon retrieval of the last response results are combined and returned to the frontend. Since the search APIs differ slightly (i.e. document search suports pagination while concept search does not) the client creates one request for each of the result domains (concepts, documents, statements). Whenever results retrieved for a domain the respective section of the GUI is updated immediately. Each search results displays the title of the result. More information can be obtained by clicking on "More Info".

The search interface for triples returns RDF from the selected data sources. Given that most biomedical experts are not familiar with Semantic Web technologies, we opted for presenting a verbalized version of the triples as title. The triples themselves are presented when more information is requested by the user. The verbalization of the triples was realized by extending the SPARQL2NL framework and is presented in Section 3.4.

3.2.3 Data Selection

Data selection is performed directly from the search results screen. The user interface as depicted in Figure 3.4 for each search result exhibits two buttons:

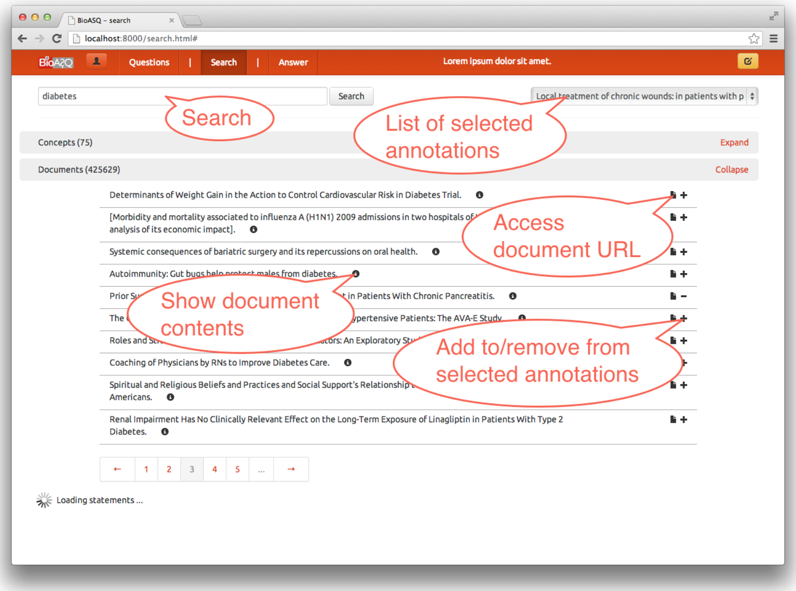
- a document icon that can be used to view the original resource (i.e. document URL),
- a plus sign that, when clicked, adds the respective resource to a list of resources marked to be used during annotation.

Each of the selected resources is made available as annotation in the next step.

3.2.4 Annotation

The annotation process itself is split into three substeps:

1. Formulating the answer.
2. Annotating the answer with selected documents or parts of them.
3. Storing the complete answer on server.


BioASQ - search
localhost:8000/search.html#
BioASQ
Questions
Search
Answer
Lorem ipsum dolor sit amet.
Search
List of selected annotations
Access document URL
Show document contents
Add to/remove from selected annotations
Expand
Collapse
← 1 2 3 4 5 ... →

diabetes

Local treatment of chronic wounds: in patients with p

Concepts (75)

Documents (425629)

Determinants of Weight Gain in the Action to Control Cardiovascular Risk in Diabetes Trial.

[Morbidity and mortality associated to influenza A (H1N1) 2009 admissions in two hospitals of
analysis of its economic impact].

Systemic consequences of bariatric surgery and its repercussions on oral health.

Autoimmunity: Gut bugs help protect males from diabetes.

Prior Su … in Patients With Chronic Pancreatitis.

The … ypertensive Patients: The AVA-E Study

Roles and St … tors: An Exploratory St

Coaching of Physicians by RNs to Improve Diabetes Care.

Spiritual and Religious Beliefs and Practices and Social Support's Relationship
Americans.

Renal Impairment Has No Clinically Relevant Effect on the Long-Term Exposure of Linagliptin in Patients With Type 2 Diabetes.

Loading statements ...

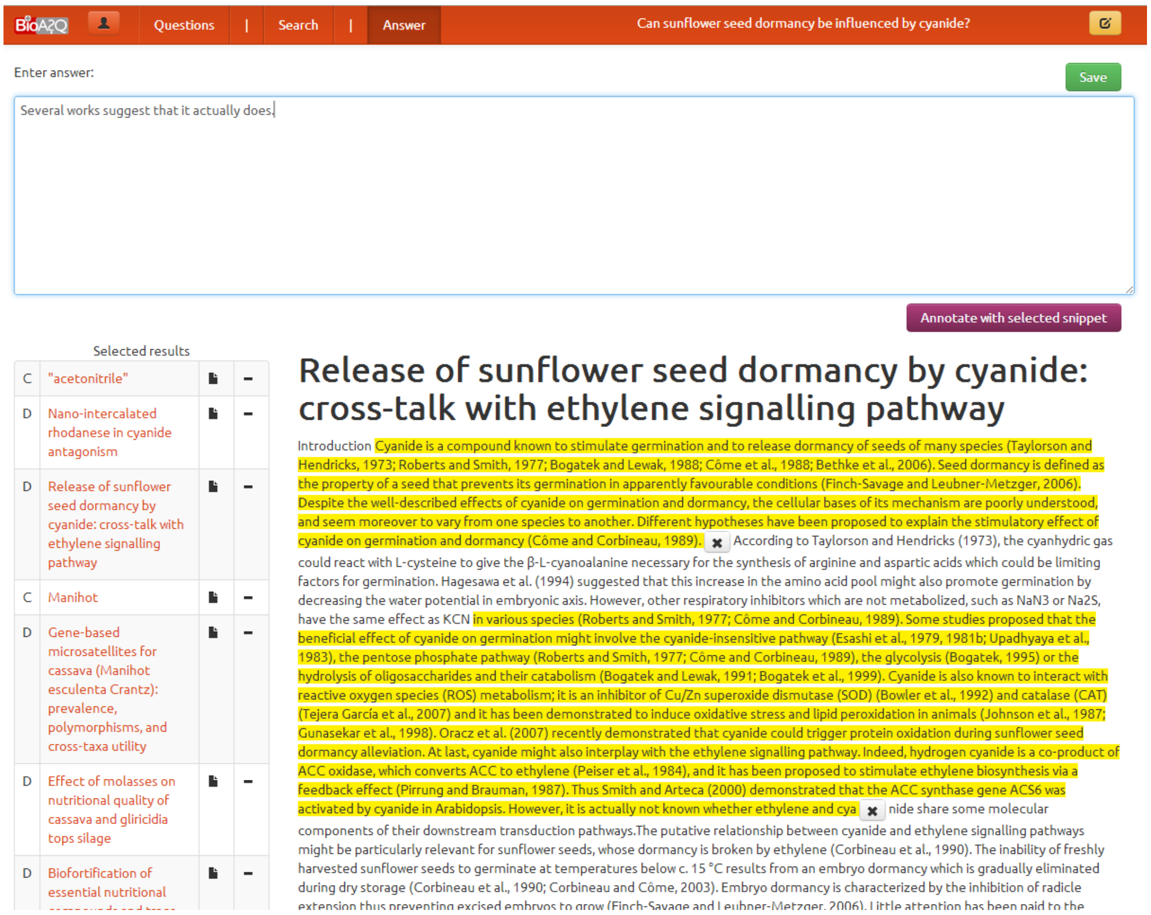
Figure 3.4: Screenshot of the annotation tool's search and data selection screen with the section for *document results* expanded.

The first step is the formulation of the answer to selected question by the domain expert. After the careful formulation of the answer, the said answer can be annotated with the documents, snippets, concepts and triples that were selected at the search stage. The annotation is carried out as depicted in Figure 3.5. For the first version of the challenge, we assume that all documents, concepts and triples selected by the user during the search phase are used to annotate the whole answer. The core of the annotation process here is thus either to dismiss items that were selected in the previous step or to add snippets (i.e., document fragments) as annotations to the answer. The deletion of items can be carried out by simply clicking on the $-$ icon next to the items that are to be deleted. The addition of snippet is similarly simple: The user simply has to select the document from which he/she wishes to use snippets for annotation. Then, he/she selects the fragment from the document and clicks on "Annotate with selected snippet". Selected snippets can be deleted at any time by simply clicking on the $\times$ icon at their bottom right end. The annotation is considered completed when the user clicks on the "Save" button at the upper right of the annotation window. This action stores all annotations in the database on the server side. Obviously, saving can be carried out as often as needed. Moreover, all annotations carried out by the user are also stored locally in the browser cache. Note that the user can go back to the search or question formulation tab at any time. Therewith, the tool ensures maximal annotation flexiblity.

3.2.5 Storage

The data storage implemented by our tool relies on the NoSQL database MongoDB. We chose this framework because it supports a very flexible data model resembling RDF and has an open-source nature and high scalability. The data is stored internally according to the schema presented in Figure 3.6.

The core entities stored by the tool are questions, answers and annotations. We assume that each *question* has exactly one *answer* and that each answer is annotated by *snippets*, *documents*, *concepts* or *triples*. Each question is created by one user (the *creator*) and is associated to a *type*, which states


BioA2Q
Questions
Search
Answer
Can sunflower seed dormancy be influenced by cyanide?
Save
Annotate with selected snippet

Enter answer:

Several works suggest that it actually does.

Selected results

C	"acetonitrile"	🗎	−
D	Nano-intercalated rhodanese in cyanide antagonism	🗎	−
D	Release of sunflower seed dormancy by cyanide: cross-talk with ethylene signalling pathway	🗎	−
C	Manihot	🗎	−
D	Gene-based microsatellites for cassava (Manihot esculenta Crantz): prevalence, polymorphisms, and cross-taxa utility	🗎	−
D	Effect of molasses on nutritional quality of cassava and gliricidia tops silage	🗎	−
D	Biofortification of essential nutritional	🗎	−

Release of sunflower seed dormancy by cyanide: cross-talk with ethylene signalling pathway

Introduction Cyanide is a compound known to stimulate germination and to release dormancy of seeds of many species (Taylorson and Hendricks, 1973; Roberts and Smith, 1977; Bogatek and Lewak, 1988; Côme et al., 1988; Bethke et al., 2006). Seed dormancy is defined as the property of a seed that prevents its germination in apparently favourable conditions (Finch-Savage and Leubner-Metzger, 2006). Despite the well-described effects of cyanide on germination and dormancy, the cellular bases of its mechanism are poorly understood, and seem moreover to vary from one species to another. Different hypotheses have been proposed to explain the stimulatory effect of cyanide on germination and dormancy (Côme and Corbineau, 1989). ✖ According to Taylorson and Hendricks (1973), the cyanhydric gas could react with L-cysteine to give the β-L-cyanoalanine necessary for the synthesis of arginine and aspartic acids which could be limiting factors for germination. Hagesawa et al. (1994) suggested that this increase in the amino acid pool might also promote germination by decreasing the water potential in embryonic axis. However, other respiratory inhibitors which are not metabolized, such as NaN3 or Na2S, have the same effect as KCN in various species (Roberts and Smith, 1977; Côme and Corbineau, 1989). Some studies proposed that the beneficial effect of cyanide on germination might involve the cyanide-insensitive pathway (Esashi et al., 1979, 1981b; Upadhyaya et al., 1983), the pentose phosphate pathway (Roberts and Smith, 1977; Côme and Corbineau, 1989), the glycolysis (Bogatek, 1995) or the hydrolysis of oligosaccharides and their catabolism (Bogatek and Lewak, 1991; Bogatek et al., 1999). Cyanide is also known to interact with reactive oxygen species (ROS) metabolism; it is an inhibitor of Cu/Zn superoxide dismutase (SOD) (Bowler et al., 1992) and catalase (CAT) (Tejera García et al., 2007) and it has been demonstrated to induce oxidative stress and lipid peroxidation in animals (Johnson et al., 1987; Gunasekar et al., 1998). Oracz et al. (2007) recently demonstrated that cyanide could trigger protein oxidation during sunflower seed dormancy alleviation. At last, cyanide might also interplay with the ethylene signalling pathway. Indeed, hydrogen cyanide is a co-product of ACC oxidase, which converts ACC to ethylene (Peiser et al., 1984), and it has been proposed to stimulate ethylene biosynthesis via a feedback effect (Pirrung and Brauman, 1987). Thus Smith and Arteca (2000) demonstrated that the ACC synthase gene ACS6 was activated by cyanide in Arabidopsis. However, it is actually not known whether ethylene and cya ✖ nide share some molecular components of their downstream transduction pathways.The putative relationship between cyanide and ethylene signalling pathways might be particularly relevant for sunflower seeds, whose dormancy is broken by ethylene (Corbineau et al., 1990). The inability of freshly harvested sunflower seeds to germinate at temperatures below c. 15 °C results from an embryo dormancy which is gradually eliminated during dry storage (Corbineau et al., 1990; Corbineau and Côme, 2003). Embryo dormancy is characterized by the inhibition of radicle extension thus preventing excised embryos to grow (Finch-Savage and Leubner-Metzger, 2006). Little attention has been paid to the

Figure 3.5: Screenshot of the annotation tool's answer annotation process

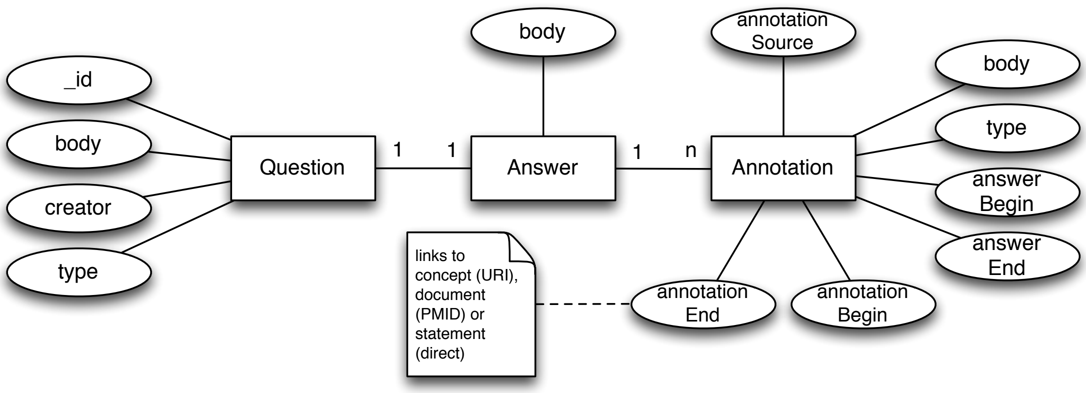


Figure 3.6: Entity-relationship diagram of the data schema used to store questions, answers and annotations.

the awaited type of answers. Currently, the tool supports 4 main types of questions: Yes/No, factoid, list, and summary. All answers have a body (which is basically text), of which each fragment can be annotated.

3.3 Search Modules

For the purposes of *BioASQ* tasks 1a and 2a, we have indexed the *PubMed* data, e.g., approximately 23 million entries with titles and abstracts of the papers, as well as the *MeSH* ontology, from which the class labels (*MeSH* headings) are drawn. Regarding tasks 1b and 2b, in addition to the aforementioned, we have indexed:

- the *Jochem* ontology, for the purpose of covering drugs
- the *UniProt* database (the *Swiss-Prot* component), for the purpose of covering targets
- the *Gene Ontology*, also for the purpose of covering targets
- the *Disease Ontology*, for the purpose of covering diseases
- the *LinkedLifeData* triples, for the purpose of covering the needs of tasks 1b and 2b regarding extracted facts (approximately 8 billion statements, which also include all of the statements extracted from *UMLS*)
- approximately $800,000$ full text articles from *PubMed Central*, for the purpose of expanding the *PubMed* document source with searchable full text articles. The articles are offered through a particular agreement with *TI*.

All of the aforementioned resources have been indexed by *TI* and are provided through respective Web services. The ontological resources have been converted to proper *OBO* files, i.e., files formatted following the *OBO Foundry* Flat File Format Specification for ontologies[4]. The concept names (labels), their synonyms and their relations have been indexed in separate *Lucene* indexes. For the document resources, also *Lucene* indexes are used, applying the standard *Lucene* analyzer for the English language.

The *API* through which the resources may be accessed, is based on *JSON*. For each resource, a respective service is implemented in a unique *URL*. Each *URL* request opens a session and may request the results, given a query, e.g., a concept, using *HTTP-POST* and a parameter *json*. The reply (the value of the *json* parameter) is a *JSON* object that contains the results for the given query. In the case of the ontological resources, the result list contains concepts from the respective ontology, and in the case of the document sources, the result list contains citations from *Medline* (title, and abstract), or full text articles from *PubMed Central*.

A list of the services that have been developed follows, with a short description of the input and output parameters used for accessing the resources and getting results.

- In the URL: `http://www.gopubmed.org/web/bioasq/mesh/json`, a service for accessing the *MeSH* ontology, with input parameter "*findEntity*", and output parameter "*findings*", which contains the list of related concepts (a list of "*concept*" entries with "*label*" entries), given the query submitted with the input parameter. Additional information is provided inside each "*label*" entry in the *JSON* object, such as "*termId*" and "*uri*" of the concept. In addition, inside each "*concept*" entry, the offsets in which the query keywords matched each returned concept are provided.

[4] `http://www.obofoundry.org/`

- In the URL: `http://www.gopubmed.org/web/bioasq/go/json`, a service for accessing the *GO* ontology, with the same input and output parameters as aforementioned.
- In the URL: `http://www.gopubmed.org/web/bioasq/uniprot/json`, a service for accessing the *UniProt* database, with the same input and output parameters as aforementioned.
- In the URL: `http://www.gopubmed.org/web/bioasq/jochem/json`, a service for accessing the *Jochem* ontology, with the same input and output parameters as aforementioned.
- In the URL: `http://www.gopubmed.org/web/bioasq/doid/json`, a service for accessing the *Disease Ontology*, with the same input and output parameters as aforementioned.
- In the URL: `http://www.gopubmed.org/web/bioasq/pubmed`, a service for accessing the *PubMed* indexed documents (titles and abstracts), with the same input parameters as aforementioned, and the output parameter containing "*documents*" entries in the returned *JSON* object. Each entry has a "*pmid*" element, which is the *PubMed* id of the indexed citation, a "*documentAbstract*" entry, and a "*title*" entry. In addition, the *MeSH* annotations are provided when available.
- In the URL: `http://www.gopubmed.org/web/bioasq/pmc/json`, a service for accessing the *PMC* full text articles, with the same input parameters and output parameters as aforementioned, with the only difference being that the articles returned contain in addition the full text.
- In the URL: `http://www.gopubmed.org/web/bioasq/linkedlifedata/triples`, a service for accessing the *LinkedLifeData* platform triples. The input parameter is "*findTriples*", and accepts any keywords as query. The output parameter contains a list of "*triples*" entries. Each entry has in turn a "*subj*", "*pred*", "*obj*" and "*score*" field, representing the subject, the predicate and the object of the triple, and the matching score given the input query.

3.4 Verbalization Service

The verbalization service was implemented by extending the SPARQL2NL framework developed by ULEI.[5] The approach followed by SPARQL2NL is tailored towards SPARQL constructs typically used in keyword search and question answering, and it consists of four main steps: a *preprocessing* step which normalizes the query and extracts type information for the occurring variables, a *processing* step during which a generic representation of the query is generated, a *postprocessing* step which applies reduction and replacement rules in order to improve the legibility of the verbalization, and a *realization* step which generates the final natural language representation of the query. As an exemplary use case, SPARQL2NL has been integrated into a user interface for the question answering system TBSL to enable users to read and disambiguate the different SPARQL queries generated when processing a question.[6]

The rationale behind the service is that RDF triples can be regarded as variable-free atomic graph patterns. Thus, the SPARQL2NL approach can be used to verbalize RDF triples. The realization of an RDF triple `s p o` depends mostly on the verbalization of the predicate `p`. If `p` can be realized as a noun phrase, then a possessive clause can be used to express the semantics of `s p o`, as shown in 1. For example, if `p` is a relational noun like `classis`, then the verbalization is `?x's classis is ?y`. In case `p`'s realization is a verb, then the triple can be verbalized as given in 2. For example, if `p` is the verbal expression `is part of`, then the verbalization is `?x is part of ?y`.

[5] `http://aksw.org/Projects/SPARQL2NL.html`

[6] A demo can be found at `http://autosparql-tbsl.dl-learner.org`.

1. ρ`(s p o)` $\Rightarrow$ `poss(`ρ`(p),`ρ`(s))` $\wedge$ `subj(BE,`ρ`(p))` $\wedge$ `dobj(BE,`ρ`(o))`
2. ρ`(s p o)` $\Rightarrow$ `subj(`ρ`(p),`ρ`(s))` $\wedge$ `dobj(`ρ`(p),`ρ`(o))`

Note that `BE` stands for the verb "to be" and ρ is the realization function. In order to automatically determine which realization to use, we relied on the insight that the first and last word of a property label are often the key to determining the type of the property: properties whose label begins with a verb (resp. noun or gerund) are most to be realized as verbs (resp. nouns). We devised a set of rules to capture this behavior:

- Property labels which begin by a past tense hint toward verbal clauses.[7]
- Property labels which begin by a gerund and end in a noun hint toward possessive clauses.
- Property labels which begin with a noun are usually noun phrases.
- Property labels which begin with a verb are usually verbs.

In some cases none of the rules apply. In these cases, we compare the probability of $P(p|\texttt{noun})$ and $P(p|\texttt{verb})$ by measuring

$$P(p|X) = \frac{\sum\limits_{t \in synset(p|X)} \log_2(f(t))}{\sum\limits_{t' \in synset(p)} \log_2(f(t'))}, \tag{3.1}$$

where $synset(p)$ is the set of all Wordnet synsets of p, $synset(p|X)$ is the set of all synsets of p that are of the syntactic class $X \in \{\texttt{noun}, \texttt{verb}\}$ and $f(t)$ is the frequency of use of p in the sense of the synset t according to WordNet. For

$$P(p|\texttt{noun}) \geq \theta \times P(p|\texttt{verb}), \tag{3.2}$$

we choose to realize `p` as a noun; else we realized it as a verb. We evaluated the approach on the QALD-2 question answering benchmark as shown in Table 3.1. The verbalization service [8] requires three variables (strings):

- the label of the subject (`subject`),
- the label of the predicate (`predicate`),
- the label of the object (`object`).

The exemplary call `http://139.18.2.164:9998/verbalizer?subject=Manioc&predicate=contain&object=cyanide` returns {`"Manioc contains cyanide."`}.

[7]Note that such word forms can be easily detected in English, while infinitive and third person forms are often identical with noun forms.

[8]Accessible at `http://139.18.2.164/verbalizer/`.

Dataset	Namespace	Frequency	#Verbs	#Nouns	Accuracy in %		
					$\theta = 1$	$\theta = 2$	Baseline
DBpedia-test	`property`	40	8	25	87.50	**90.00**	75.00
	`ontology`	97	7	48	91.75	**94.85**	86.60
	Other	99	2	1	**98.99**	**98.99**	32.32
	Overall	236	17	74	94.07	**95.76**	61.86
DBpedia-train	`property`	41	1	26	**100.00**	**100.00**	80.25
	`ontology`	81	5	43	95.06	**100.00**	85.37
	Other	135	3	2	**98.51**	**98.51**	42.96
	Overall	257	9	71	97.67	**99.22**	61.48

Table 3.1: Accuracy of realization of atomic graph patterns. Namespace stands for the namespace of the properties used in a SPARQL query. Frequency denotes the number of times that a property from a given namespace was used, for example `property` (which stands for `http://dbpedia.org/property`) or `ontology` (`http://dbpedia.org/ontology`). #Verbs (resp. #Nouns) is the number of properties that were classified as verbs (resp. nouns).

Bibliography

N. Heino, R. Speck, T. Ermilov, and A.-C. Ngonga Ngomo. BioAsq Annotation Tool GIT, January 2013. URL `https://github.com/AKSW/BioASQ-AT`.

A.-C. Ngonga Ngomo, L. Bühmann, C. Unger, J. Lehmann, and D. Gerber. SPARQL2NL GIT, January 2013. URL `https://github.com/AKSW/SPARQL2NL`.