RNDP

IDP Institute for Data and Process Science

Research Notes on Data and Process Science

September 2022 Issue 3 ISSN 2702-508X
DOI: 10.57688/334

Article

Ontology-based Semantic Matching for Technology Transfer

S. Weber, S. Berg, A. Khelil, T. Lehner, A. Chebaane, B. Bilgin, K. Ramadani, C. Doering

The Institute for Data and Process Science (IDP) is a research institute of the University of applied Sciences Landshut, Germany.

The Research Notes on Data and Process Science is published digitally at least twice a year by the IDP. Responsibility for the content rests upon the authors and not upon the IDP.

All articles are peer reviewed by at least two reviewers.

Impressum / Imprint

Research Notes on Data and Process Sciences (RNDP)

Editors: Prof. Dr. Holger Timinger
Prof. Dr. Maren Martens
Prof. Dr. Abdelmajid Khelil
Prof. Dr. Mona Riemenschneider

ISSN: 2702-508X

Contact: Institute for Data and Process Science
University of Applied Sciences Landshut
Am Lurzenhof 1
D-84036 Landshut
Germany

Web address: http://idp.institute

DOI: 10.57688/334

Ontology-based Semantic Matching for Technology Transfer

Stefan Weber
Institute for Data and Process Science
University of Applied Science Landshut
Am Lurzenhof 1, D-84036 Landshut, Germany

Simeon Berg
Institute for Data and Process Science
University of Applied Science Landshut
Am Lurzenhof 1, D-84036 Landshut, Germany

Abdelmajid Khelil
Institute for Data and Process Science
University of Applied Science Landshut
Am Lurzenhof 1, D-84036 Landshut, Germany
ORCID: 0000-0002-4536-8058

Thomas Lehner
Institute for Data and Process Science
University of Applied Science Landshut
Am Lurzenhof 1, D-84036 Landshut, Germany

Ahmed Chebaane
Institute for Data and Process Science
University of Applied Science Landshut
Am Lurzenhof 1, D-84036 Landshut, Germany
ORCID: 0000-0003-0525-1207

Berkin Bilgin
Institute for Data and Process Science
University of Applied Science Landshut
Am Lurzenhof 1, D-84036 Landshut, Germany

Kreshnik Ramadani
Institute for Data and Process Science
University of Applied Science Landshut
Am Lurzenhof 1, D-84036 Landshut, Germany

Claudia Doering
Institute for Data and Process Science
University of Applied Science Landshut
Am Lurzenhof 1, D-84036 Landshut, Germany
ORCID: 0000-0002-3727-8773

Abstract

Recently, Technology Transfer plays a crucial role in transferring knowledge and technologies from academia to the industry and society through a centralized web repository. However, due to the large volume of the data transfer and lack of quality and inconsistency, the search engine may not easily be able to understand the user's interests that lead to insufficient and inappropriate matching results. This article proposes a novel ontology-based semantic web tool for universities to match collaboration requests to the most suitable available competencies. The proposed tool highly automates the matching process for a timely matching results without sacrificing their quality/accuracy.

1 Introduction

A third mission for universities in the form of sustainable transfer of knowledge and technology from and to industry and society is becoming increasingly important alongside the standard teaching and research missions [**?**]. The theoretical framework for this transfer was already described in the 1980s and can be found in the concepts of "Entrepreneurial Universities" [**?**], "Triple Helix" [**?**] and "Mode 2" [**?**]. All of these concepts incorporate the idea that universities should actively interact with their surroundings to benefit all stakeholders. Nevertheless, transfer often still takes place in a spontaneous manner without strategic coordination. In addition, the digital transformation of transfer activities is still in its infancy. Therefore, the right support structures need to be in place to allow for an easy and accessible handling of these transfer and third mission activities. A common transfer portal, in which transfer and cooperation requests are processed, can simplify the collaboration between all partners.

Researchers in universities can usually indicate their research or collaboration interests through a website profile or internal discussion with the transfer center, which will then start a matching process to find suitable companies/social stakeholders or other researchers. External partners may express their collaboration interest through submitting a request to the transfer center of a university. The transfer center then forwards the request to potential researchers and waits for a prescribed time period for a response. In case of positive response, the transfer center interlinks the researchers with the collaboration partner. Obviously, this approach is resource and time consuming, which may lead to a loss of opportunity to cooperate. Accordingly, the digitalization of this matching process is crucial for universities to sustain within a promising transfer situation.

For instance, a digital knowledge/expertise/needs matching portal could significantly foster timely bringing together interested parties for collaboration and transfer. Such a matching portal is primarily intended to give innovative companies or stakeholders from society, the opportunity to network with the competencies from universities that are important to them and to quickly find out about available ser-

vices and offers.

In this article, we present a novel web-based transfer portal for universities and industry/society to match collaboration requests with existing competencies. This matching/transfer portal provides industry/society partners with recommendations of promising research partners, which are suggested to them based on their managed digital profile as well as their individual search queries for cooperation partners (requests). There are two possible approaches to automate the matching. First, a syntactic search engine can be used. This is based on the query of each user. An advanced variant is the semantic search engine. In this article, we present a novel concept that enriches the search capabilities through semantic-aware matching based on ontologies.

In the matching process, the semantic distance between the search key word and the researcher fields of expertise are taken into account. For example, when searching for 'web development', researchers from the faculty of computer science are prioritized, as the underlying ontology models 'web development' as subordinate to 'computer science'. Thus, the system can assign specific research areas of researchers to the respective subject areas. This functionality is realized by the semantic data model. Using ontologies, the application generates a hierarchy that helps the algorithm to identify the suitability of a researcher for collaboration with the company.

Obviously, a rich hierarchical ontology repository significantly improves the quality of query results. Therefore, an API service is provided to allow uploading of new ontologies to the portal ontology repository. In addition, we provide a web application that allows non-IT-affine users to create ontologies that are compatible with the ontology repository.

We structure the remainder of this article as follows. In Section 2, we briefly present important basics, in particular, the definition of semantic and ontologies. In section 3, we discuss the state of the art. In Section 4, the requirements are highlighted. In Section 5, the ontology-based semantic matching approach is detailed. In Section 6, we revisit our requirements and show to which extent the proposed solution fulfills them. Section 8 concludes the article and presents relevant future work.

2 Basics

In the following, we briefly define semantics and ontologies.

2.1 Semantic Definition

The Semantic Web is the next evolutionary stage of the World Wide Web (WWW). In what so-called Web 3.0, not only is information linked together, but web content is also enriched and networked by machine-readable semantic metadata. It optimizes the exchange of information on the Web by allowing machines to distinguish machine-readable meanings (semantic content) and process them in a targeted manner. Entities are used to create references that occur in a particular context (ontology). This allows the context and meaning of the entity to be identified. Entity can thus be classified, but also evaluated. it can be uniquely identified by a combination of a unique ID and certain properties. For example, terms with multiple meanings can be uniquely identified depending on their meaning, such as "jaguar" (jaguar can mean animal, tank as well as car). it can be a person, company, fact and abstract things as well as building. In semantics, ontologies represent an environment in which entities can be related to each other. A specific industry can serve as an ontology in which different market participants/companies act as entities and are related to each other.

2.2 Ontology Definition

In computer science, ontologies describe a set of objects and their properties or semantic relationships to each other, which are described in a standardized languages [**?**]. One of these languages is the Resource Description Framework (RDF). This allows information to be passed on in the form of RDF triples. Triples consist of a subject, to which the triple refers, a predicate, which represents a property or a connection, and an object, the target of the predicate. Each element in the triple has a unique Internationalized Resource Identifier (IRI), with which it can be found by other elements. In summary, ontologies represent a set of information, represented in RDF triples, in a form readable by machines. These triples can also be rendered graphically and are thus human readable. Ontologies are used in computer science, for example, in the area of databases, as in this project with semantic database search, or in the area of artificial intelligence. The ontologies are used to create a hierarchy of domains. The hierarchy is then used to perform the semantic search by drawing inferences and capturing relationships. Furthermore, the ontologies may contain additional information about the subjects, for example, information about the type of language, synonyms, etc.

Ontologies usually have the following structure: Individuals (instances), classes (concepts), attributes, and relations.

There are several tools to edit ontologies. The most popular one is Protégé [**?**]. Simple Knowledge Organization System (SKOS) is a W3C standard, based on other Semantic Web standards (RDF and OWL), that provides a way to represent controlled vocabularies, taxonomies and thesauri.

3 State of the Art

Several successful matching web portals do exists. Some of them target online dating. Others focus on matching the offering/ advertisement to the user's profile. For matching research expertise, there exist also commercial web portals such as those that match free lancers to project offering. To the best of our knowledge there is no matching tool that focuses on technology transfer for universities.

Search engine is a crucial technique for WWW or internet as it supports users to search and fetch the desired information based on a specified query. Information can be explored based on two approaches: syntactic search including Google, Yahoo, Wikipedia, etc or semantic search including Kosmix, Hakia, Swoogle, DuckDuckGo, etc [**?**]. In this section, we focus on the related work of the semantic web search engines based on Resource Description Framework (RDF) [**?**] and Ontology Web Language (OWL) [**?**]. Paramjeet et al. [**?**] proposed a novel approach using ontology for semantic linking and storing open government datasets of agriculture, land and rainfall sectors in New Zealand. Kalampokis et al. [**?**] developed a linked data architecture based

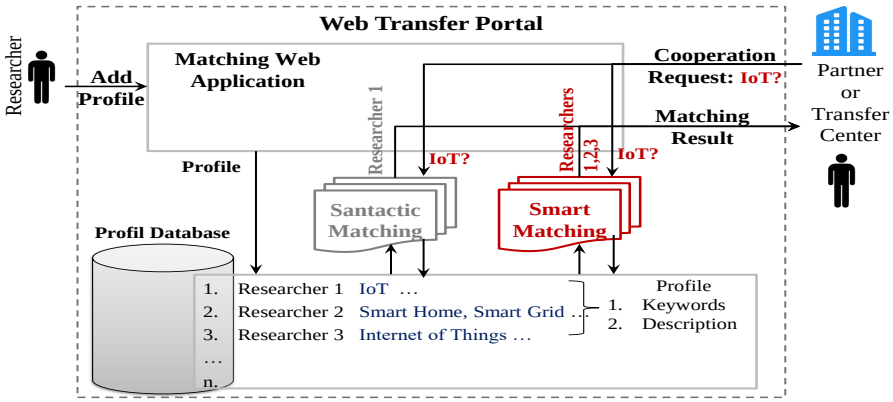

Figure 1: Web Application System Architecture [Matching Request: IoT; Researcher Profile: Researcher 1: IoT — Researcher 2: Smart Home, Smart Grid — Researcher 3: Internet of Things]

on OWL SameAs links [**?**] between Uniform Resource Identifiers (URIs) to connect different Open Government Data (OGD). Three participants were involved in the dataset. In addition, SPARQL [**?**] protocol and RDF Query Language are used to access the designed dataset. A search engine model is proposed by Jiang et al. [**?**] to link the concept of transportation domain ontology manually and automatically. In [**?**] the authors provide a new approach for Valencia's water resources management by exploiting and publishing ontology-based systems using linked open data. However, the existing semantic searches still require improvement such as low system performance due to the long term process of the proposed techniques.

In this article, we are the first who target the data transfer ontology domains among universities and companies. In addition, we enable users (scientists in our case) to build and upload their ontologies on our proposed system that are not considered in the previous research.

4 Requirements

The users of the web transfer tool had some requirements which were recorded and specified. Then they were categorized into two groups: Those defined for the user interface and ontology management, and those for matching based on ontologies. The requirements were then sorted according to the order of prioritization of the users.

4.1 User Interface and Ontology Management

Requirement 1 - Implementation as a web application: This allows the application to be accessible platform and location independent.
Requirement 2 - External uploading: The web portal must allow uploading of externally created ontologies.
Requirement 3 - Creation of tree-based ontologies: The creation of simple, tree-based ontologies must be made possible for the user. In addition, the creation should be uncomplicated and no previous knowledge about ontologies should be necessary.
Requirement 4 - Overview Page: The application should have an overview page, where the already existing ontologies are presented. The users have the possibility to download created ontologies in order to edit them with an external tool.
Requirement 5 - Scalability: In order to support a large number of users from multiple universities and multiple enterprises, the software need to be scalable.
Requirement 6 - Configurability: The application must be configurable on-the-fly without interruption. In particular updating the ontology repository should be conducted without interrupting the application.
Requirement 7 - Editing rules: Editing or deleting the uploaded ontologies via the web application must be disabled. Instead, a modified version of them can be uploaded in addition.
Requirement 8 - Templates for creating: The creation of simple ontologies can be facilitated using the publication of already created ontologies as templates.
Requirement 9 - Security: The general security of the web application and the API should be ensured as well as possible.

4.2 Matching

Requirement 10 - Results of searching: The search should return only relevant results.
Requirement 11 - Relevance: Consequently, results should be sorted according to their relevance.
Requirement 12 - Logical operators: With the help of logical operators, the searcher shall be given more possibilities to formulate the query.
Requirement 13 - Search terms: Since it may not be possible to clearly determine what the entrepreneur is looking for on the basis of the search terms, it would make sense to offer suggestions to the entrepreneur. This would ensure a refinement of the search. Categorizations can be used for this purpose. For example, it can be offered here to search in specific faculties or departments. These suggestions are created based on the ontologies.

5 Ontology-based Semantic Matching

Our solution is composed of two building blocks: Matching algorithm and ontology management.

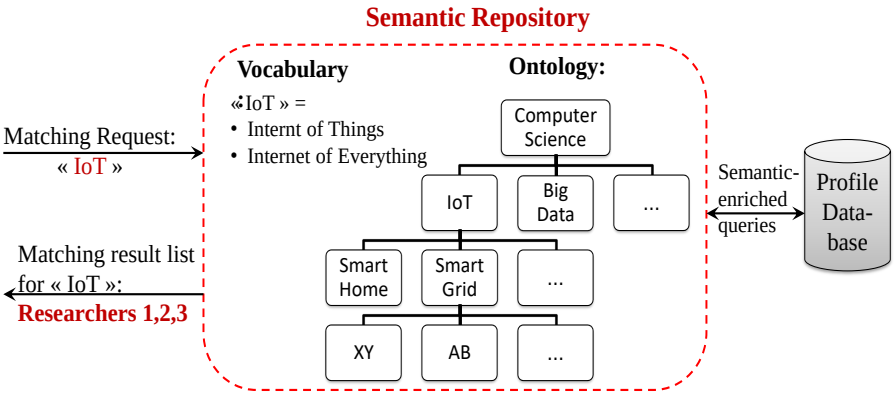

Figure 2: Web Application System Architecture (Matching Request: IoT; Researcher Profile: Researcher 1: IoT — Researcher 2: Smart Home, Smart Grid — Researcher 3: Internet of Things)

5.1 Overview of Approach

First, the system architecture is going to be illustrated. Next, we briefly outline the structure of simple ontologies. Then, the saving process of ontologies is explained.

5.1.1 System Architecture

Figure 1 illustrates the system architecture. This system provides various services that can be called by the web application, e.g., ontology upload service. To allow the integration of a MySQL database the mysql package was used. Cross-Origin Resource Sharing (CORS) [**?**] requests are possible because of the cors package. The setup of the API application is done as already described in the setup of the web application. In the following, the settings and services of the application are described in more detail.

To meet the requirements for configurability the API is also configurable. These settings can be made in the settings.json, this must be created by copying and renaming the template-settings.json. The following is a description of the asynchronous services provided by the API.

Our objective is to develop a smart matching in the web portal. First, the researchers add their profiles in the web portal database. Next, partners or the university transfer center send a matching request through the web portal to find the list of researchers, e.g., who are working on Internet of Things (IoT) as an example. A query sent from the smart matching to the database to match IoT in the profiles fields mainly in keyword and description. Finally, the matching result (a list of most suitable researchers) is returned to the requester Figure 2.

Matching is basically a search operation. The syntactic search is a simple search based on a query and check if the matching request exists in the researcher profile or not, in this case the matching result contains only Researcher 1. Though “Smart Home” and “Smart Grid” are subdomains of IoT and “Internet of things” is the same as IoT, Researchers 2 and 3 are not proposed for the cooperation. We propose to use semantic search based on a semantic repository that includes vocabularies and ontologies in order to optimize the accuracy of matching. A vocabulary contains synonyms and implies that Researcher 3 is added to the matching result list. An ontology is based on graphs, for example IoT has a “Smart Home” and ”Smart Grid” as subdomains. Accordingly, Researcher 2 is inserted to the matching result list.

5.1.2 Tree Structure for Simple Ontologies

The SKOS is automatically added to the header when the ontology is uploaded. This ensures that only ontologies and namespaces that are actually needed are imported. Since this only happens in the background during the upload, the user does not notice this step.

In the web editor, adding new NamedIndividuals is very simplified. Users only have to press a plus button, either at the root of the hierarchy or a branch of it. A new individual is then added as a child of the selected parent. The user does not have to worry about assigning the has broader properties. Changing the parent can also be done simply by drag and drop. A disadvantage, however, is that it is not possible to give a child several parents due to the tree structure. This possibility already exists in Protégé, but it bears the risk that users accidentally introduce a recursion into their ontology, which the algorithm in the transfer portal recognizes as an error.

The web editor also only offers the option of labels that can really be read and thus ensures compatibility with the transfer portal.

In summary, the web editor is faster and safer than Protégé. Experienced users should also find the editor pleasant, as they do not have to adhere to rigid rules.

5.1.3 Ontology Saving

At this point there are hardly any differences between the editors. In both, the ontology can be saved locally so that it can be worked on later. The only difference is in the handling of uploading finished ontologies; in Protégé this has to be done via the ontology browser, whereas in the web editor it can be uploaded directly by clicking on the ”Upload” button.

5.2 Matching Algorithm Based on Ontologies

The web application was realized as a Node.js application and uses the JavaScript framework ReactJS. Node.js has the advantage that the setup and extension of the project is fast and straightforward, which makes the application easy to scale. ReactJS allows to divide the JavaScript code into many small components, which makes the code more manageable. Additionally, it allows components to be reused. The requirements were also taken into account when developing the ontology editor, so it was required that the creation

is simple and clear and the created ontologies are compatible with the algorithm. The page of the ontology editor instantiates the class ontology and stores it in a state, which is passed as prop to the components of the layout.

Here it will be shown how the matching algorithm is working in a real-world example Figure 3.

5.3 Management of Ontology Repository

In the following, we briefly explain the methods for the interfaces used for data transmission and data binding.

5.3.1 Get /get/user/:userID

This service is no longer used in the current web application. In order to provide an example for possible future work based on this work, this service has been retained in the code, but it is disabled by the ”enable_get_user” setting. The method gets the ID it is looking for as a request parameter and searches the database for a user with this ID and returns its data to the client. Otherwise the status ”400” is transmitted.

5.3.2 Get /list/ontologies

This service returns a list of all ontologies stored in the repository. Two folders must be specified in the settings. These serve as storage locations of the simple and professional ontologies. Both folders are iterated over and all files that have the correct extension (.json / .ttl / .rdf / .owl) are stored as an object in an array associated with the folder. This object contains the name of the file, its extension, the size on the server and the path. Both arrays are returned to the client bundled in a JSON object.

5.3.3 POST /get/ontology

This service can be used to download ontologies stored on the server from the client. For this, a form with two fields is passed to the service, which are then read using the formidable package. These fields describe the name of the ontology and its type (”simple” or ”advanced”). Then, depending on the type, an ontology with the given name is searched in the respective folder and sent to the client. If no ontology with the requested name is found in the folder, or generally wrong parameters were passed, the service responds with the status ”400”.

5.3.4 POST /upload/ontology/advanced

This service allows uploading an ontology to the server. For this, a form with two fields is passed to the service. These fields describe the name of the uploaded file and the file itself. The formidable package is used to read these fields. The new path of the uploaded ontology is then created from the name and the specified folder. If the file extension is not present in the allowed extensions (.rdf, .owl, .ttl) or if a file with the same path already exists, the process is aborted and the status ”400” is sent. In case of a successful upload, the status ”200” is returned to the client.

5.3.5 POST /upload/ontology/simple

This service can be used to upload the ontologies created by the client in the web editor and their templates. The submitted form again contains a field for the file name and fields for the template file in JSON format and the ontology created by the client in Turtle format. From the file name, paths for the submitted files are again generated, where the template gets the path of the ”simple” folder, while the ontology is stored in the ”advanced” folder. If an error is thrown during the upload, for example because one of the files already exists, the upload of both files is aborted and the client is sent the status ”400”.

5.3.6 Security

The services for uploading the ontologies could pose a risk, as executable files could be uploaded here. Therefore, these services check the file extension of the uploaded files before saving and abort if it is not accepted.

It would be possible for the service to read an ontology (”POST /get/ontology”) to send server files to an attacker if the attacker specifies a path outside the API in the passing argument (e.g.: ”../../var/backups”). To prevent this, only the folder (”simple” or ”advanced”) and the file name can be passed to this service and the path of the file is created by the service itself. Thus only files in the above mentioned folders can be read.

The ”GET /list/ontologies” service can be used to perform so-called ”amplified reflection” DDoS attacks. In this case, a user calls the service and forges its address so that the data is sent to the attack target. Since the request consumes less memory than the response, the target can be blocked without overloading the attacker’s own resources.

In order to increase the security of the application, a ”rate-limiter” was included here, which limits the number of calls to the services for each IP address. Users can thus only send a certain number of requests to the API per time unit, after which all calls are blocked.

After describing the implementation and function of the application in the previous sections, this section is briefly devoted to proving the compatibility of the ontologies with the transfer portal algorithm. The application of the transfer portal was installed on a local computer and then an ontology created by the web editor was moved into the folder [./ontologies] of the transfer portal application. As an example ontology the ontology Informatik.ttl was used. After that, the transfer portal application was started via the start.bat script. The application then iterates over all ontologies located in the ./ontologies folder and generates a list of all subject areas that were found. If errors occur during the evaluation, they are displayed in the console window. After the algorithm ran successfully, the site.html page was opened in the browser. On this page there is an input field which retrieves the list of topics from the server and displays them in a dropdown. The display of the dropdown may take some time, since both the execution of the algorithm and the retrieval of the list are asynchronous. The individual topics can be selected and are then listed below the input field. The algorithm must be adapted to the application. A parameter for the restriction to relevant search results was added. The constraint parameter can be increased as the search results may still be too broad. However, this can be better decided once the

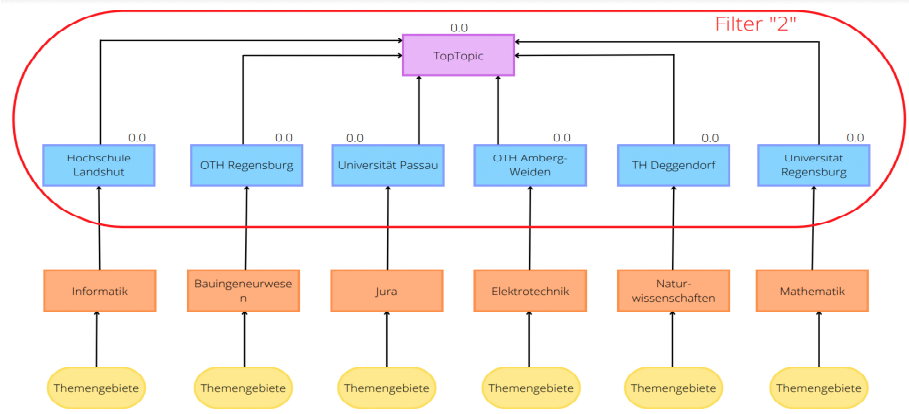

Figure 3: Matching Algorithm in the way of Solution Finding

actual ontologies are available. The requirements for the ontologies, but also the structural arrangement plays a crucial role here. The framework that is defined in the requirements for the ontologies has strong similarities to the trees in data structures. Possibly, the algorithm could be applied to such as well. However, the advantage of ontologies is the flexibility and diversity of semantic data structures. Also, information can be added to the semantic data structure without limiting the function of the algorithm.

6 Evaluation

After all registered requirements were integrated into the web transfer tool, they were tested and evaluated from the creators of the tool. In the following they are described briefly.

Requirement 1: Implementation as a web application - The application is a Node.js application based on ReactJS. The application is passed from the server to the client and runs in the user's browser.
Requirement 2: Uploading externally created ontologies - Uploading ontologies created with external editors is possible through the ontology browser.
Requirement 3: Creation of simple ontologies - The creation of new ontologies is possible via the ontology editor. The comparison between the generation in the web editor and the creation in Protégé indicates that this process is simpler and clearer in the ontology editor. The proof of compatibility has been shown before.
Requirement 4: Implementation of an overview page - The overview page was implemented as an ontology browser. This allows, as required, the upload and download of all already uploaded ontologies. Existing ontologies are displayed in two tables.
Requirement 5: Scalability - Both applications were built using Node.js. The setup of the application is very simple. In this regard, the use of React components makes it even easier to extend the application.
Requirement 6: Configurability - Configurability was enabled by integrating multiple settings files. The texts in the frontend can also be changed using translation files.
Requirement 7 and 8: Editing Rules and Templates for Creating - Rules for editing and functions for creating templates are given and developed in a good way.
Requirement 9: Security of the ontologies - As mentioned in the description of the API services, they only allow saving ontologies if no ontology with the same name already exists. Therefore, it is not possible to delete or overwrite existing ontologies. Among the requirements, which are considered as nice to have, the following are the creation of new ontologies based on templates. The selection of an existing ontology as a template has been implemented in the development of the ontology browser. Here the user can select an ontology and is then redirected to the editor. This opens a new ontology with the settings of the template. Additionally, the bonus includes the security of the application. As described in the assumptions, the security of the web application was trusted to be provided by the server. Authentication of users must already happen in the transfer portal. During the implementation of the API, attention was paid to the security of the services.
Requirements 10-13: Due to variations in execution time, it is difficult to make statements about the reliability of the search algorithm. The structure or size of the ontology can only be estimated and simulated. These factors have a significant impact on the execution time. The following assertions can be made: After repeating the search query, the search results are displayed within an acceptable time. The application can still be optimized if the execution time during execution is not suitable.

7 Discussion

As it is the case with most research, this work also has some limitations. The initiation and implementation of transfer is still a topic that both occupies and challenges universities. Although this work and the created transfer web tool proposes a solution for transfer activities, it is still just a starting point for further development. Digitalization itself is a major challenge for higher educational institutions and therefore this transfer web tool needs to be further evaluated with future users and in different university settings. As the web tool if currently only provided with a university collaboration in Bavaria, Germany, it would be necessary to also evaluate it within other universities or other collaborations.

8 Conclusion

The digitalization of the technology transfer process and in particular of the matching of cooperation partners is crucial.

As syntactic matching suffers from low accuracy, we proposed a novel solution for semantic matching. Our key contribution is a web tool that allows the management and the creation of simple ontologies that represent the knowledge basis for reasoning about the semantic, i.e., the meaning of text elements. A simple creation of ontologies is fundamental to motivate researchers without prior knowledge to create ontologies about their research field and upload them to the web transfer portal. Experienced users, can however use their preferred tools to create ontologies for the transfer portal.

In future work, we aim to extend the tool in order to fully support Classes and ObjectProperties beyond NamedIndividuals for ontolgies.