# A Semantic Question Answering in the Domain of Smart Factories

Orcun Oruc, Adrian Singer, Ken Wenzel, *Fraunhofer IWU, IEEE*

**Abstract:** The industrial manufacturing changed its characteristics by the increasing interconnection of machines and devices (e.g. Industrial Internet of Things, IIoT). Smart factories have been evolved complex linked data aspect of manufacturing devices over the past few years. Leveraging *Industry 4.0*, smart factories require a machine-to-human or machine-to-machine cooperation to present information to experts or human operators. The Human Machine Interaction (HMI) devices mostly use semantically created data provided by production systems. Nowadays, smart factories produce a large amount of data that need to be apprehensible by HMI Devices. OPC UA is a de facto standard to tackle the data source problem; several proposals introduced relevant to the question answering. The researcher proposes to develop a question answering that serves as an industrial assistance process and assess the efficiency and accuracy of it.
Consequently, a semantic question answering that has been placed in a smart factory can use streaming data generated by production systems, statically generated data by OPC UA Servers or Semantic Tools such as eniLINK [1]. Nature of the semantic question answering might have different properties from open domain or closed domain question answering so that the researcher will examine a restricted domain question answering system with linked data through natural language processing architecture. Moreover, the researcher presents a semantic question answering with semantic data which are based on the eniLINK and generated data from an OPC UA Server known as *Dynamic Server*. Lastly, the researcher will evaluate the semantic question answering with different measurements according to the requirements of a restricted-domain question answering.

***Index Terms*— Semantic Web, Web 2.0, Web Services, Information Retrieval, Industry 4.0**

## I. INTRODUCTION

THIS work introduces a new concept for smart factories in terms of linked data processing integrated into a semantic question answering. The Semantic Web is a state-of-the-art research area that handles the semantical understanding data between human to machines and machine to machines. In a smart factory, connected sensors, actuators or manufacturing devices creates a massive amount of data.
Thus, a vast amount of unlabeled data could not be used by applications, so that *W3C (World Wide Web Consortium)* decided to create a standard for Semantic Web in order to apply linked open data concept. Fraunhofer IWU started to design its smart factories that are capable of generating structured linked data. A semantic question answering is used for information retrieval to provide answers from questions through linked data. The proposed semantic question answering can understand complex natural language expressions, and it can respond to the user by answers. Mainly, the semantic question answering system employs unstructured data or structured data. The researcher takes linked data generated by an OPC UA Server that is named *DynamicServer* and the *eniLINK* streaming data. The empirical analysis indicates the answer return rate and precision; therefore, it evaluates the usability for a human operator, experts or an end-user of a web application.
The goal of this research is to show a model of semantic question answering for a smart factory that utilizes the natural language expressions as sentences, questions or keywords to give a precise and rapid answer to human operators or experts.
The remainder of this paper was structured as follows: Section II will provide a brief overview of previous studies about algorithms for the question answering and evaluation criterion. Section III summarizes the research approach of the semantic question answering aspect of the smart factory constructed by Fraunhofer IWU. Section IV explains the state-of-the-art status of *Industry 4.0* and *Smart Factories*. Section V introduces the serialization process from the *Information Model* and from streaming data to linked data. Section VI and VII introduces theoretical background and practical implementation respectively. As for Section VIII, the researcher will explain the test environment; accordingly, the researcher gives the results of the semantic question answering. Next, the researcher will answer specified research questions in order to clarify key points in Section IX. Consequently, the researcher concludes in Section X.

## II. RELATED WORKS

[Molla, Vicedo 2007] reviewed a primary characteristic of question answering in restricted domains is the integration of domain-specific information that is either developed for question answering or disclosed for other purposes [2]. [Molla, Vicedo 2007] defined main characteristics of question answering system over limited domains, e.g. circumscription of question answering, the complexity of question answering, and practical usage of question answering [2].

The authors have compared between open-domain and restricted-domain question answering by figuring out key points. [Molla, Vicedo 2007] offers four clear-cut subjects such as *size of data*, *domain context*, *resources*, and *use of domain-specific resources*.

[Ferre 2012] published one of the detailed reports that express common pitfalls of natural language processing, and essential points while consolidating SPARQL query and morphological definitions [3]. SQUALL is a solution for querying and updating RDF graphs by exploiting controlled natural language expressions that restrict grammar structures of a sentence in order to diminish complexities [3]. It has been grouped all substantial features of a morphological language, and the author pointed out what type of features in a natural language harnesses with regarding priorities and orders. The main contribution of SQUALL is categorizing ambiguities of natural expressions and how turned an advantage out when using a controlled natural language [3].

[Biswas, Sharan & Malik 2014] proposed an architecture that extracts precise answers for a given question [4]. The authors described the module distinctly and defined the types of questions that can be asked against the question answering.

The authors sketched a translation from their intermediate language to SPARQL to gain more accuracy with their system [3]. Template based solutions were commented for a restricted domain and open domain question answering systems. [Unger Et al. 2012] proposed a template based solution that produce a SPARQL template, which it directly mirrors the internal structure of the question [5]

Evaluation of a semantic question answering is still a cumbersome and hard problem. Lack of test questions that belong to specific domain is one of the major problems. [Diekerma, Yılmazel & D. Liddy 2004][6] offer different methodology from an open-domain question answering while evaluating the restricted domain question answerıng. The authors specify the evaluation methodology as below [6]:

**System Performance**: Speed and Availability

**Answers**: Accuracy, Completeness

**Display User Interface**: Querying styles, natural language queries, keywords, browsing, and the Question Formulation Assistance (Spell Checker, Abbreviation Solver)

The authors stated that the *TREC* style question answering evaluation might not be suited to their restricted domain system so that user-based evaluation can be more viable in order to evaluate the system [6].

## III. RESEARCH APPROACH

**Research Questions**:
*RQ-1 Can a semantic question answering utilize heterogeneous linked data source (e.g., OPC UA Information Model, streaming data, static data) in the domain of smart factory?*
*RQ-2 What are the requirements of the Semantic Question Answering for smart factories?*
*RQ-3 What are the main features associated with the methods of the Semantic Question Answering*
*RQ-4 Can the researcher generalize our approach to other plants of and how did the research contribute to the research area?*

**RQ-1**: Today, a smart factory creates a massive amount of data by leveraging big data analysis technology. However, the data source suffers from comprehensible by applications. The research question relates to the implementation of a serialization process into linked data. This research question evaluates the types of data source by implementing solutions.
**RQ-2**: The research question relates to algorithm design thinking and domain-specific requirements to fulfill information retrieval theory and natural language understanding. This question has to evaluate the practical application.
**RQ-3**: This research question assesses the pros and cons of our approach and gives the list of features of a semantic question answering in the domain of smart factory.
**RQ-4:** The research question examines the viability of the proposal in an aspect of division of a plant or a smart factory. Generated new test parameters set to evaluate our semantic question answering. In the test phase, the researcher will see how to create the questions.

## IV. SMART FACTORIES AND INDUSTRY 4.0

The definition of smart factories has been unfolded over the past few years. In the present study, a smart factory was defined an aspect of boosted technologies named *Industry 4.0* and *Human-Machine Interaction*. Improvements in manufacturing development have affected economic growth over the last few decades in Germany. Continuous improvement of *Industry 4.0* brought the researchers to find cutting-edge technologies such as question answering systems, manufacturing augmented reality, and so on.

A smart factory is a highly digitized and connected production facility that relies on smart manufacturing [7] — this concept is one of the key outcomes of Industry 4.0, which it intelligently changes manufacturing technologies. Smart manufacturing is a term coined by a set of departments of the United States [8]. The central power of the smart factory is that it makes data collection possible. Additionally, sensors enable the monitoring of specific processes throughout the factory that increases awareness of what is happening on multiple levels [9].

The development of *Industry 4.0* has a significant influence on the manufacturing industry. In the era of smart manufacturing systems, *Industry 4.0* needs to standardize all connection pipelines in smart factories. The primary objectives of Industry 4.0 are that making the manufacturing technologies of factories more intelligent, optimizing the chain of processes, and enhancing capabilities of communication one to another. Moreover, *Industry 4.0* enforces end-to-end digital integration of engineering throughout the value chain to facilitate highly customized products, thus reducing internal operating costs [10].

## V. LINKED DATA SERIALIZATION

The data sources of the researcher are structured as semantic data source. All data sources have linked triples regardless of the type of semantic data, such as Turtle, RDF or OWL.

### *A. The Meaning of Data for OPC Unified Architecture Information Model*

OPC Unified Architecture was developed for devices of industrial internet of things to remedy problems about *service orientation*, *loose coupling*, and *object-orientation paradigm*. The OPC UA has been evolved starting from OPC to OPC UA over the past few decades, and architectural design entirely was changed. The OPC was dependent on the *Component Object Model* that should work with only Microsoft documents. The fundamental restriction of OPC was that it was restricting devices to connect just to a Windows-based operating system, and there was no service orientation. After developing the *Distributed OPC* and *OPC UA* idea, the foundation of OPC has constructed a viable concept for object-oriented, loose coupling and service orientation in a manufacturing system.

Aside the OPC UA is a complex protocol; the OPC UA is one of the ubiquitous industrial communication protocol that can be used in the various stage of the manufacturing. Thanks to the OPC client-server architecture, any devices can connect to the protocol in a manufacturing system. A programmable logic controller, a sensor or an actuator can connect to the same server, and they can assign their values into different folder organizations to represent data in an address space. The address space is a data plane for an OPC UA Server; hence it should coordinate *variables*, *methods*, *objects*, and *nodes* respectively. An end-user can identify primitive and user-defined types so that the complex structure of devices can be represented as a whole in a big data plane. However, this data plane only provides definitions and types.
The *Information Model* supports object-oriented paradigm such as abstraction and inheritance between *References* and *Objects*. It is well known that an object can live as a *Node Class* in the address space. The objects may have relationships with other objects in the information model. Utilizing *References*, a user can traverse in the address space of OPC UA to reach all level of nodes and variables. Nevertheless, neither the address space nor the information model is far apart from understanding the meaning of data. Semantic understanding of the OPC UA Information Model has a vital role in building up an answering question system. The information model holds all device-specific information such as *device type*, *data changes of the device*, *vendor type* and *relationship between devices*. These information sources would be helpful to a human operator or expert who works with manufacturing systems.

### *B. Mapping an OPC UA Data into a Semantic Data*

The primary data sources are semantically parsed data from *eniLINK* [11] and the OPC UA Server in Fraunhofer IWU named *Dynamic Server*. In the phase of OPC UA Server generated data, the researcher used an SDK which is published by FreeOPCUA [12][13]. The researcher contributed to [12][13] with extra conversion steps such as XSLT and triple store processing.
OPC UA Standard utilizes an information model and the information model can be used with metadata to simulate in other OPC UA Servers with languages such as XML (Extensible Markup Language). Due to the nature of XML, it is a language that depends on strong hierarchical elements and hardly extendable. However, semantic data such as Resource Description Framework (RDF) can employ triples with the SPARQL query language. Different RDF Graphs are stored in eniLINK[11] are uniquely identified.

**Algorithm 1** Node Extraction

```
function MainFunction()                                ▷ Starting point
    export = ServerExport(serverurl, filename)
    export.IMPORT NODES(serverurl)
    export.EXPORT FILE(outputFile, namespaces)
    export.XSLTCaller()
function Build Node Tree(nodes)                        ▷ Node Formatting
    client ← GetEndpoint()
    client ← Client(serverurl)
    global nodecumulated ←None
    nodeID ←0
    for node in nodes do
        nodecumulated = node.nodeid.Namespaceindex
        for ref < node.getreferences() do
            nodecumulated.extend( ref.nodeid.Namespaceindex)
        nodecumulated = list(set(nodecumulated))       ▷ Clear duplicates
    return nodecumulated                               ▷ Return node id list
function Import Nodes(serverurl)                       ▷ Traverse Node
    client = Client(serverurl)
    client.connect()
    for ns < client.getNamespaces() do
        namespaces[client.getNamespaceIndex(ns)] = ns
    root = client.getRootNode()
    child = client.iterateChildNodes(root)             ▷ append child to node
function Export File(outputFile, namespaces = None)    ▷ Export into XML
    if namespaces != None then
        for node not in nodes do
            if node.nodeid.namespaceindex is namespaces then
                nodes = [node]
            else
                nodes = list(nodes)

    export = XmlExport(client)
    export.BUILD NODE TREE(nodes)
    export.appendXML(outputFile)
```

**Figure V-1.** OPC UA Information Model Serialization

The algorithm, as shown in Figure V-1, identifies tree elements of a node by taking namespace indexes. The *namespace index* contains *node ids*. Once a user performed to browse from a node to another, the user needs to know node identification number. If the user did not range the total number of references, the application should get all nodes that have references until the algorithm reaches all elements of the mesh network. Accumulated nodes were inserted into a list to export an XML format. After obtaining XML structures, the system can convert the elements into linked data such as the Turtle through *Extensible Stylesheet Language Transformations*. XSLT can transform from the XML format to RDF format by minimizing the blank nodes. Once the system converted to RDF/XML format, graph libraries can deal with the conversion process into triple formats. The application only take cares the uniform locator identifier to do a conversion, and then the application ought to arrange uniform locators by considering them different from 'example.org'.

### *A. Real Time Data Mapping with KVIN*

Real-time data sources have intricacies in the use of linked data taken from sensors, actuators or software logs. In the aspect of smart factories, sensors and actuators that are the underlying structure of manufacturing machines mostly create continuous streamed data. Fraunhofer IWU collects the real data source by saving into a time series database. The major drawback was about that when the time series data were taken, the endpoint of a semantic query cannot use the unstructured data without annotating it. Such annotations can be the formulation of triples, insertion of predicates or serialization from one formal language to another.
The proposed architecture provides a real-time semantic data annotator KVIN that utilizes to extract triples from time-series data in a database.

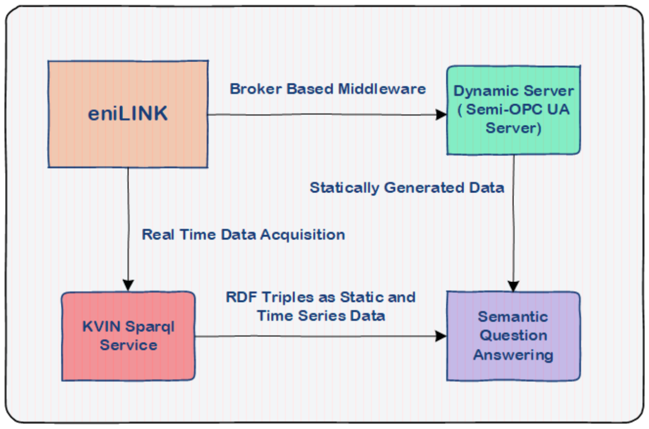

eniLINK
Broker Based Middleware
Dynamic Server ( Semi-OPC UA Server)
Statically Generated Data
Real Time Data Acquisition
RDF Triples as Static and Time Series Data
KVIN Sparql Service
Semantic Question Answering

**Figure V-2.** KVIN Continuous SPARQL Mapping

This work proposes a service named *KVIN* to perform a SPARQL request against a specified endpoint. This service is based on a combination of the triple store through the *Level DB* which is a key-value storage library written by the Google company[1]. It has been used an RDF4J's extension to create a SPARQL Service. After obtaining time series data, the data are mapping the SPARQL triples. These graphs contain mapped triples with their time-stamped and value in order that the researcher can employ values with some complex process with SPARQL language. Moreover, a federated service replies to the queries determined by users, which it reduces the answer return time of a question answering system. The KVIN does not create instantly hard-coded triples or a new language such as C-SPARQL. It only arranges the size of the time window and puts the graphs into the service to present to the end user.

## VI. THEORY OF THE NATURAL LANGUAGE UNDERSTANDING

In natural language processing, the researcher needs to identify the sentence's structure in order to reach the step of Query Formulation. The following methods that the researcher used in the practical application are concisely given.

**Semantic Query Language:** SPARQL was designed the use of semantically structured triples, not for relational datasets. *PREFIX*, *SELECT* and *WHERE* are three basic operators of SPARQL Protocols. *PREFIX* makes the serialization steps easier in referencing to the *Uniform Locator*. *UNION* statement can help at federating multiple triples into a single query — the *OPTIONAL* statement is

[1] https://github.com/google/leveldb

used to allocate a particular portion of SPARQL into triples. As can be seen, the primary goal of the query language federating information among heterogeneous systems.

**Preprocessing and Tokenization**: Chiefly, all natural language processing tasks start with preprocessing, which means that cleaning the data for specific tasks could be the reduction of non-optimized data and discrepancies between the values or removing non-related morphological properties. A question answering system should parse natural language expressions as tokens. Tokenization is an initial step for the part-of-speech tagging to parse into a verb, noun, cardinal numbers, or an adjective and so on.

**Lemmatization and Stemming**: *Lemmatization* and *Stemming* are similar steps to each other with one difference. While a stemming algorithm is used to find syntactical structures, a lemmatization algorithm looks for a semantic structure. Stemming clears the morphological structure of suffix and prefixes. In this proposed system, the researcher is supposed to use a lemmatizer and stemmer in order to reduce lexical complexities. A lemmatizer is used to consider the morphological analysis of verbs, e.g. from *"contains"* or *"contained"* to *"contain"*. Then the researcher takes this verb to map into a predicate to construct a SPARQL Query. The lemmatization and stemming are part of the normalization process in terms of morphological properties.

**Part-of-Speech Tagging**: It is a preprocessing step for parse trees to identify item taggers such as verbs, adjectives or nouns. A sentence consists of a couple of structure, including expressions like nouns, verbs, pronouns, prepositions, adverbs, conjunctions, participles and articles that are main categories of part of speech processing [14]. The part-of-speech tagger mostly applies the *Markov Model* [14] that is a part of statistical natural language understanding. The *Markov Model* stands for a state can depend on a previous step, but there is no dependency on states of historical steps more than one. For instance, a noun or a verb defines its neighbors, e.g. nouns are preceded by determiners, adjectives, verbs [14]. As an example, a chess player moves any chess piece according to the last movement of a rival rather than guessing from the first movement of the competitor. In this way, pre-saved corpora that have a massive amount of words have to be tagged by the POS Tagger.

**Parsing:** The approach of parsing äs divided into two main sections, which are the rule-based approach and the probabilistic approach [15]. The rule-based approach is a top-down approach to solve problems via predefined rules such as regex-parsing and character-based parsing. Therefore, a question answering system should define rules precisely to get the correct answer. Open-domain question answering systems use this approach because of the high complexity of the bottom-up approach and broaden question types. Nevertheless, the rule-based approach could give undesirable results in restricted-domain question answering so that this could be a time-wasting and an error-prone approach.
A dependency parser analyzes the grammatical structure of a sentence, and it gives the relationship among them. The dependency parser also gives the relationship between dependent words and root words. Thus, the researcher can identify the center verbs or dependent nouns of complex sentences. This parser utilizes a dependency treebank file and word embedding files. Chiefly, a dependency parser applies the supervised machine learning method to reach a syntactical and semantical result.
A constituency (phrase) parser is likely known as a phrase parser that has an objective is to check the grammatical structure of sentences by parsing the chunks of morphological structure. The constituency parser may not handle the relationship among language items. Dependency parser analyzes the grammatical structure of given natural expressions to identify the relationship between a root word and dependent words which relates to the root word.

**Named-Entity Recognition**: It is a subtask of information extraction to locate and classify named entities with pre-classified labels, such as names of people, organizations, locations, etc. Named-entity recognition is a method that identifies the item of a sentence as a domain-specific. It solves the problem of recognition in the same way that the chunking method does. However, named-entity recognition may be trained with labeled data and it is more sophisticated technique than the chunking, which has the deep and shallow parsing methods.

**Similarity Analysis:** Sentence similarity is used to compare two string inputs to achieve indicative questions like *"Is the system health good?"*. Mainly, this similarity method leverages averaging word vectors such as *word2vec* and *glove* that implements *Euclidian* and *Manhattan Distances* or *Cosine Similarity*. Three similarity methods that the researcher analyzed, which shown as below:

The *Levenshtein Distance* is that the calculation time could be $O(|s1| \times |s2|)$ using $O(\min(|s1|, |s2|))$ space. After calculating distance among *s1* and *s2*, the result may be divided into maximum length of string [16]. The *Jaro Winkler* has a transposition matrix *t* with common characters that are calculated together to reach similarity value [16]. The *Jaccard Similarity* algorithm takes into consideration the size of the intersection divided by the size of the union of two sets [16]. Under the same test data and methods, similarity levels of the Jaccard, Jaro Winkler, and Levenshtein are 0.8095, 0.7544, and 0.58 respectively. The higher score shows a better performance for similarity measurement.

In order to calculate word-based similarity, the researcher uses WordNet with glove vectors. Such vectors are pre-calculated synset values were stored into a file can be downloadable. These synset values show the similarity value with cosine similarity algorithm. The WordNet can calculate the similarity of acronym and hypernym except for synonym. Calculation of semantic similarity is a hard and complicated process. As the researcher explained in the following scenario, two phrases such as 'Internet of Things' and *"Mesh Network"* are semantically similar. One of them implies *"the network of physical objects with electronics, software, sensors, and connectivity"* and the latter implies *"the topology of a network whose components are all connected directly to every other component"*. The researcher cannot easily calculate this semantic similarity. Instead of calculating semantic similarity, the researcher can calculate word vectors of verbs and nouns that relate to similarity synset. If a computed synset value is above than the threshold value, a question answering can accept these two strings similarly constructed. In the practical implementation, the researcher has used verb synonym similarity to map onto <*IRI: predicate*> sets.

**Question Classification:** Questions should be categorized to get the correct answer. It is a part of question processing that can parse the question input and assign into the correct labels. Machine learning methods can define derivation of an expected answer. This paper utilized *Logistic Regression* and *Support Vector Machine* for question classification phase. While the support vector machine was classifying the question with *TREC Dataset* [2], the logistic regression was examined the type of question at the Github repositories [17] and [18]. Questions are grouped with coarse-grained labels, which are *Abbreviation*, *Entity*, *Description*, *Human*, *Location*, and *Numeric*.
On the other hand, another dataset that the researcher has trained with Logistic Regression that comprises of 'what', 'quantity', 'who', 'unknown', and 'why' labels. The *Logistic Regression* and Linear *Support Vector Classification* have supervised machine-learning methods by identifying coarse-grained question indicators with pre-trained labels. Logistic Regression estimates the parameter with a logistic function. The type of regression allows classifying to multi-labels the afore-mentioned labels.
The *Support Vector Machine* aims to improve quality of hyperplane that separates multi-class labels. *Linear SVC* is such a method that implements a linear kernel function through the *Support Vector Machine*. The *Newton-cg* has a gradient descent function that reduces the error rate each iteration to find out global minimum. The Limited BFGS is an optimization method over the Newton-cg. Logistic Regression Cross Validation (CV) applies cross-validation to training and test set by splitting at particular percentages between them. The result has been listed in Table VI.1.

| Parameters | Precision | F1 | Recall |
|---|---|---|---|
| Newton-cg | %95.55 | %95.56 | %95.57 |
| Linear SVC | %92.75 | %92.76 | %92.77 |
| Limited BFGS | %94.21 | %94.22 | %94.23 |
| Logistic Regression CV | %95.63 | %95.63 | %95.64 |
| Linear SVC for Li&Roth Taxonomy | %65 | %45.5 | %35 |

**Table VI.1:** The evaluation of the Question Classification

[2] https://trec.nist.gov/data.html

## VII. PROPOSED ARCHITECTURE OF THE SEMANTIC QUESTION ANSWERING

The researcher is implementing a mixed parsing based approach in order to define essential elements of a natural query. The major priority is to detect <*subject-predicate-object*> triples and then mapping the verbs and nouns onto template SPARQL. This template was created according to the requirements of a smart factory. For instance, dynamic queries that fetch information from streaming data possibly need *SUM*, *AVG*, or *MIN* filter statement of SPARQL language.
As for static queries, the researcher has hierarchical data and semantical data of the Information Model. Listing VII-1 shows an example hierarchical triple of the smart factory of Fraunhofer IWU. Such predicates <*factory: contains*> should be parsed, and they need to be matched with verbs. However, this may lead the researcher to a misconception to match synonym verb of predicates. Therefore, as illustrated in Figure VII-1, the researcher has an extra step to identify the synonym of verbs.

```
<http://linkedfactory.iwu.fraunhofer.de/linkedfact
ory/linkedfactory/demofactory/machine10>
factory:contains
<http://linkedfactory.iwu.fraunhofer.de/linkedfact
ory/demofactory/machine10/sensor1>
```

**Listing VII-1:** Sample triples of eniLINK

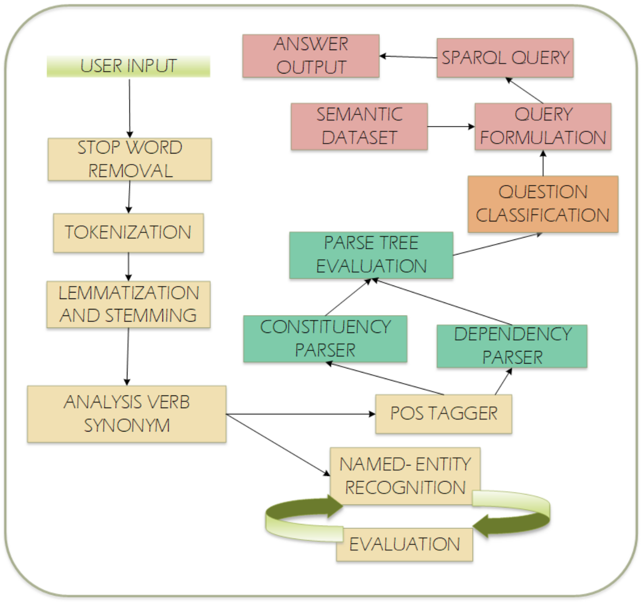


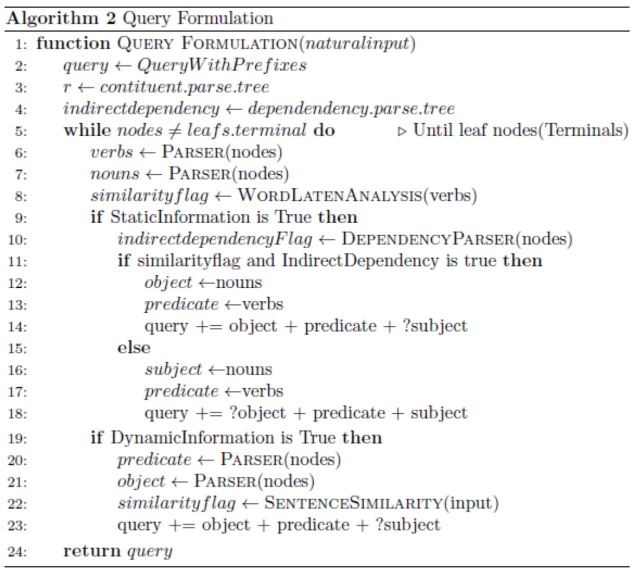
**Figure VII-1.** Natural Language Processing Steps for Question Answering

After taking input from any user, stop-word preprocessing stage start to filter unnecessary characters such as question marks, exclamation points, commas, dots or determiners.

Tokenization is the next step to reduce the size of characters to provide optimization in natural language processing, and it reduces the complexity of instances of sequence characters. Lemmatization and stemming are fundamental steps before WordNet verb analysis since the primary target is to extract verb, nouns and related chunking in order to formulate a SPARQL query that can give an answer.

There is a control step for the named-entity recognition after finding synonyms of the verb. As previously explained, it is a way of extracting most common entities such as location or names. Names, locations or organizations can face a problem about identifying domain-specific objects. For instance, linkedfactory can be comprehensible for Fraunhofer IWU's smart factory, but another smart factory or different domain may not know what kind of entity is. Therefore, if the question answering can catch the entity-relationship pair as shown in Figure VII-3, the question answering system inserts natural expressions into shallow and deep parsing steps.

For dynamic queries, the question answering system applies a similarity measurement. In Figure VII-2, the similarity flag employs a sentence similarity in the following case. *"Is the system trouble"* is a reasoning query. The system should interpret this query, and the system needs to know exactly the semantic meaning of the sentence. However, the approach as above-statement is a similarity-based identification. When a user a question like *"Is the system trouble for sensor1 in machine1?"* the semantic question answering can interpret a reasoning question without understanding the underlying semantic meaning.



```
Algorithm 2 Query Formulation
function QUERY FORMULATION(naturalinput)
    query ← QueryWithPrefixes
    r ← contituent.parse.tree
    indirectdependency ← dependendency.parse.tree
    while nodes ≠ leafs.terminal do            ▷ Until leaf nodes(Terminals)
        verbs ← PARSER(nodes)
        nouns ← PARSER(nodes)
        similarityflag ← WORDLATENANALYSIS(verbs)
        if StaticInformation is True then
            indirectdependencyFlag ← DEPENDENCYPARSER(nodes)
            if similarityflag and IndirectDependency is true then
                object ←nouns
                predicate ←verbs
                query += object + predicate + ?subject
            else
                subject ←nouns
                predicate ←verbs
                query += ?object + predicate + subject
        if DynamicInformation is True then
            predicate ← PARSER(nodes)
            object ← PARSER(nodes)
            similarityflag ← SENTENCESIMILARITY(input)
            query += object + predicate + ?subject
    return query
```

**Figure VII-2.** Natural Language Processing Steps for Question Answering

The architecture has provided a SPARQL endpoint for local static data, and the KVIN presents a SPARQL Endpoint for time-series data. The researcher is using different techniques for different question types. In case of a given natural language expression as below:

**"Could you tell me which one contains fofab?"**

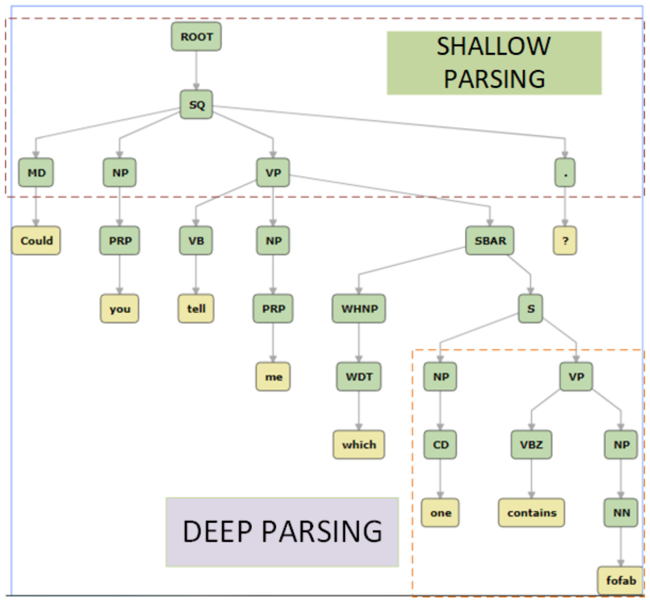


**Figure VII-3:** An example sentence from Stanford CoreNLP [19]

The researcher specifies noun and verb phrases at a basic level, which they are using a shallow parsing that can make the constituency-parsing step easier. If the system catches the right verb-noun pairs, it should eliminate expressions to reach the origin of the noun or verb. Such expressions may represent determiners, adjectives or pronouns. As shown in Figure VII-4, the system has two verbs that it needs to map the predicate of triple onto the Turtle RDF data source. If it may find out the similarity level of *'contains'* and *'tell'*, the question answering could say the essential verb to be evaluated. However, the order of a verb is important for direct and indirect questions. As shown in Figure VII-4 and Figure VII-5, multiple objects have relationships with the head verbs 'tell' and 'contains'. Subjects and objects can inverse the order of SPARQL query. In this case, the system needs to identify universal dependencies[3]. A named entity recognition can show these types of relationships as illustrated in Figure VII-6 and Figure VII-7. A drawback about this identification is a particular keyword may perplex of the identifier, noun, etc. In essence, the question answering system needs more in-depth analyzes to solve the perplexities of unique keywords and open-domain words.

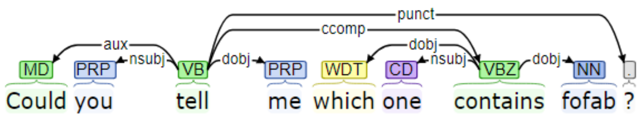

punct
ccomp
aux
nsubj
dobj
dobj
nsubj
dobj
MD PRP VB PRP WDT CD VBZ NN
Could you tell me which one contains fofab ?

**Figure VII-4:** An indirect Query Dependency Parser[19]

[3] https://universaldependencies.org/

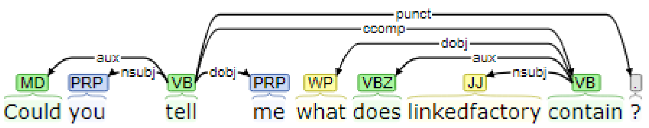


**Figure VII-5:** A direct query with Dependency Parser[19]

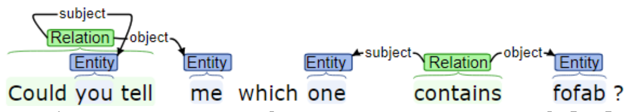


**Figure VII-6:** Named-Entity Recognition Result [19]

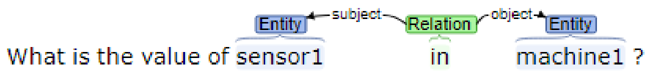


**Figure VII-7:** Named Entity Recognition with Open IE[19]

## VIII.EVALUATION

### *A. Test Environment*

In the evaluation phase, the data sources are semantic data from OPC UA, the eniLINK semantic data that consists of elements under the linkedfactory [20], and streaming data that resides in eniLINK. As previously detailed the process of serialization, the researcher has a heterogeneous data source for the semantic question answering. OPC UA Generated Data has not specific namespace definition unless the researcher defines. However, the user-defined IRIs definition has drawbacks such as collision or non-extendibility. Longlisted linked data makes the structure complex so that two subjects of the list can be overlapped because of same-defined IRIs. In this case, all namespaces are generated with http://www.example.org” or *“<unknown_namespace>”.*

The size of dataset that the researcher generated from OPC UA Server has 19, 687, which is 2 MB sized Turtle File. The Linkedfactory triples relate to hierarchical structures has 70 triples as Turtle format.

Data Sets have been specified previously so that the researcher is using the question answering with data set in a machine powered by *Intel® Core™ i7-2720QM CPU @ 2.20 GHz, 2201 MHz, and x64 based Windows 10 Pro*.

### *B. Result*

Evaluation criteria exhibit recall; accuracy, precision, and F1 score of answers against semantic question answering system. General evaluation parameters for a restricted domain question answering is not only limited with responding of questions but also the researcher can assess with speed, user interaction, querying style (keywords, browsing, spell checker, abbreviation recognition)

$$Precision = \frac{TP}{(TP + FP)} \qquad 1.1$$

$$Recall = \frac{TP}{(TP + FN)} \qquad 1.2$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{(Precision + Recall)} \qquad 1.3$$

$$Accuracy\ of\ the\ Model = \frac{(TP + TN)}{(TP + FP + FN + TN)} \qquad 1.4$$

The precision (1.1) represents expected answers was correctly predicted to the total responses. F1 Score (1.3) is a balanced weight average between the Recall and Precision. The recall (1.2) is the proportion of correctly answered questions to the total amount of questions. The accuracy of the model (1.4) shows the readers that the model was created which has a ratio of accurately predicted observation to the entire inspection.

Test questions were created with a combination of keywords and elements of sentences. Due to the domain restriction, the generation goal was answering precisely the questions ranging from keywords to complex natural input. The target data source was a mixed source that combines static and streaming data. In the appendix, one can observe combinations of test question to use further improvements.

| Evaluation Parameters | Properties |
|---|---|
| Answer Return Rate | QA against generated data from OPC UA – 23.25 seconds average<br>QA against static query from RDF file of the eniLINK – 18.92 seconds average<br>QA against dynamic query from streaming data – 17.48<br>QA against Template Based Open-Domain Questions – 20.55 seconds |
| Querying Style | Keyword-Based Search and Semantic Question Search |
| Coverage | The eniLINK data, the linkedfactory streaming data |
| Size | Static data relatively small size<br>Continuous data relatively large size |
| Up-to-dateness | No update statement provided by SPARQL |
| Query Formulation Assistance | Voice Input Recognition, Spell Checker |

**Listing VIII-1:** The semantic question answering evaluation criterion

| Question Answering Parameters | Total Questions |
|---|---|
| True Positive | 34 |
| False Negative | 13 |
| False Positive | 3 |
| Precision | %94.44 |
| Recall | %72.34 |
| F1 Score | %81.92 |
| The accuracy of the Model | %68.00 |

**Listing VIII-2:** The Evaluation of the Question Answering

## IX. DISCUSSION

In this chapter, the researcher will discuss the significance of the findings relevant to the research problem being investigated. Taking into considering the findings, the researcher will summarize insights about the problem

First, RQ-1 and RQ-2 addressed distinct architectures for the use of semantic question answering. The proposal is implementing a service called KVIN that employs key-value mapping with windowed time series data. The time series data has been windowed with the size of data that can define a scope over the data size. Although the information structure is limited to map onto Turtle triples, it can be useful for rapid prototyping. There is no cost like designing a new language onto SPARQL or overhead of instant linked data creation from streamed data.

Generating test data set still is a problematic topic for restricted domain question answering systems. For instance, test dataset for the information technology domain is not valuable for a manufacturing domain, which this restricts the testability however the researcher has used the parameters of referenced research [6]. One of the findings the answer return rate is similar to template-based open-domain question answering [21]. If the researcher wants to get an answer relevant to *node id*, *node parent id*, *references*, and the connected devices to OPC UA Server, the researcher should convert the Information Model to linked data. Converting from the root node to the leaf node with namespaces of nodes would be enough to map onto <*subject-predicate-object*> triples. The semantic question answering should give precise answers for dynamic data and list the results of answer against static data. Previous studies tried to solve the restricted domain question-answering problem with template based solution and implement a generic solution. Whereas, the researcher performed a heuristic based syntactic parsing without template-based to a particular domain.

Showing, the test results of the question answering and question classification, this study guide for the researchers of Industry 4.0 how to develop an advanced system. RQ-3 defines the main features of the semantic question answering in the smart factory domain, which is being short-listed answering, deep and shallow parsing based, and the ability to use of heterogeneous data source. Display interface may reduce the time that a human operator spends while typing and correcting spelling mistakes so that the efficiency of query processing may increase.

Consequently, as referred to RQ-4, generalization a question answering that belongs to a smart factory to other one is not an easy solution. Although the algorithm and architecture generalizations are possible; However, the drawback is the particular keywords in unstructured data and streaming data. This research contributed to the research circle with algorithms; test set generation and features of a semantic question answering to be used against heterogeneous sources.

## X.CONCLUSION

The operator assistant system increases the productivity of human operators and experts in smart factories. In this paper, the researcher has proposed an application for a restricted domain question answering that utilizes generated data from OPC Unified Architecture and streaming data. This application can reduce the total amount of time for searching through a large number of triples. The significant findings are that the proposed novel approach can be used effectively to create a supervisor tool for manufacturing technologıes and synthesized human operator assistant system that caters a robust architecture for the aimed platform. Proposed model reduced the complexity of the normalization process and employed state-of-the-art natural language understanding toolkits.

The major problem of this proposal is that the question answering solely depends on the predicates of data set that defined by the smart factory. In order to solve this problem, *subject-predicate-object* pairs can be recognized by deep learning methods with unstructured data. Correspondingly, first finding was that the named-entity recognition had shown poor performance than the parsing method aspect of identifying noun phrases and verb phrases. The second finding was that complex paragraphs need a complicated mechanism such as coreference resolution to detect an object. Speed is another factor that the researcher can infer when the point comes to customizable. Accordingly, a technical operator or expert cannot get an answer against streaming data in the constraints of a mission-critical system. The third finding was that serialization of the OPC UA is a time-consuming task; moreover, there must be a control script to detect unaltered hierarchical part. The researcher proposed the source code [22] that one could recognize simulation data in OPC UA Server with a script to stave off the repercussion while serializing .

The last finding was that implementation of a generalized algorithm could degrade the precision of answers but increase the scalability at the various department at a smart factory.

## APPENDIX

| Sample Question ID | Sample Questions |
|---|---|
| 1 | Provide me a combined result for IWU and e3sim |
| 2 | There is a member named fofab. Please give me all of its members |
| 3 | What POWERMETER holds? |
| 4 | I need to learn parent node id in generated data |
| 5 | Give me all data blocks |
| 6 | What is the value of average for the sensor1 in machine1? |
| 7 | I need to learn the maximum value of sensor5 in machine7 |

|  |  |
|---|---|
| 8 | Give me all registered node id |
| 9 | Could you tell me the system health for sensor2 in machine7? |
| 10 | Could you browse in generated data? |

## ACKNOWLEDGMENT

Fraunhofer IWU supports this work and we would like to thank the group of HMMI on the account of financial support.

## REFERENCES

[1] F. IWU, ‘eniLink’, 2015. [Online]. Available: http://platform.enilink.net/. [Accessed: 23-Nov-2018].

[2] D. Mollá and J. L. Vicedo, ‘Question answering in restricted domains: An overview’, *Comput. Linguist.*, vol. 33, no. 1, pp. 41–61, 2007.

[3] S. Ferré, ‘SQUALL: A controlled natural language for querying and updating RDF graphs’, *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7427 LNAI, pp. 11–25, 2012.

[4] P. Biswas, A. Sharan, and N. Malik, ‘A framework for restricted domain Question Answering System’, *Proc. 2014 Int. Conf. Issues Challenges Intell. Comput. Tech. ICICT 2014*, pp. 613–620, 2014.

[5] C. Unger, L. Bühmann, J. Lehmann, A.-C. Ngonga Ngomo, D. Gerber, and P. Cimiano, ‘Template-based question answering over RDF data’, *Proc. 21st Int. Conf. World Wide Web - WWW ’12*, p. 639, 2012.

[6] A. R. Diekerma and E. D. Liddy, ‘Evaluation of restricted domain Question- Answering systems’, *Cent. Nat. Lang. Process.*, pp. 12–16, 2004.

[7] R. Margaret and D. Daniel, ‘Definition of Smart Factory’. [Online]. Available: https://searcherp.techtarget.com/definition/smart-factory. [Accessed: 05-Dec-2018].

[8] K. D. Thoben, S. A. Wiesner, and T. Wuest, ‘“Industrie 4.0” and smart manufacturing-a review of research issues and application examples’, *Int. J. Autom. Technol.*, vol. 11, no. 1, pp. 4–16, 2017.

[9] C. Team, ‘What is the smart factory and its impact on manufacturing?’, *13 June 2018*. [Online]. Available: https://ottomotors.com/blog/what-is-the-smart-factory-manufacturing. [Accessed: 05-Dec-2018].

[10] T. D. Oesterreich and F. Teuteberg, ‘Understanding the implications of digitisation and automation in the context of Industry 4.0: A triangulation approach and elements of a research agenda for the construction industry’, *Comput. Ind.*, vol. 83, pp. 121–139, 2016.

[11] L. D. Platform and F. IWU, ‘eniLink’. [Online]. Available: http://platform.enilink.net/. [Accessed: 23-Nov-2018].

[12] Pure Python OPC-UA Client and Server, ‘Free OPC-UA Library’. [Online]. Available: https://github.com/FreeOpcUa/python-opcua. [Accessed: 22-Nov-2018].

[13] TU Dresden, ‘Plt-TUD’. [Online]. Available: https://github.com/plt-tud/opc_ua_xml_export_client. [Accessed: 22-Nov-2018].

[14] D. Jurafsky and J. H. Martin, ‘Speech and Language Processing’, *Speech Lang. Process. An Introd. to Nat. Lang. Process. Comput. Linguist. Speech Recognit.*, vol. 21, pp. 0–934, 2009.

[15] J. Perkins, D. Chopra, and N. Hardeniya, *Natural Language Processing : Python and NLTK*. 2016.

[16] P. Christen, ‘A Comparison of Personal Name Matching: Techniques and Practical Issues’, *Sixth IEEE Int. Conf. Data Min. - Work.*, no. September, pp. 290–294, 2006.

[17] S. Khare, ‘Question Classification Implementation’. [Online]. Available: https://github.com/swapkh91/Question-Classification. [Accessed: 26-Feb-2019].

[18] Shirish Kadam, ‘ADAM Question Answering’, 2017. [Online]. Available: https://github.com/5hirish/adam_qas. [Accessed: 12-Apr-2019].

[19] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, ‘The Stanford CoreNLP Natural Language Processing Toolkit’, *Proc. 52nd Annu. Meet. Assoc. Comput. Linguist. Syst. Demonstr.*, pp. 55–60, 2014.

[20] Fraunhofer IWU, ‘Linkedfactory Intro Page’, 2018. [Online]. Available: http://linkedfactory.iwu.fraunhofer.de/linkedfactory/view. [Accessed: 19-Feb-2019].

[21] Machinalis Group, ‘Quepy Question Answering’. [Online]. Available: http://quepy.machinalis.com/. [Accessed: 27-Feb-2019].

[22] O. Oruc, ‘Question Answering Source Code’, 2019. [Online]. Available: https://github.com/zointblackbriar/QuestionAnswering. [Accessed: 07-Apr-2019].