RFPCog: Linguistic-based Identification and Mapping of Service Requirements in Request for Proposals (RFPs) to IT Service Solutions

Hamid R. Motahari-Nezhad, Juan M. Cappi, Taiga Nakamura, Mu Qiao
IBM Almaden Research Center
San Jose, CA, United States
{motahari,jmcappi,taigamqiao}@us.ibm.com

Abstract

Request for proposal (RFP) documents describes a client’s business and service requirements in natural language. RFP packages typically consists of tens to hundreds of documents each ranging from tens to hundreds of pages and come at variety of formats and structures. Processing RFPs manually is a tedious, error prone and slow process. It is a competitive advantage to a service provider to be able to process RFP documents to automatically extract client requirements, and understand how these requirements map to the internal offerings, products or solutions of the business to improve the efficiency of preparing RFP responses, and conduct sizing and pricing of the solutions. However, this is a challenging task due to the complexity and variety of forms that requirements are expressed, their level of detail, style and language expression. In this paper, we present a novel cognitive solution that employs linguistic-based and machine learning methods for automated processing of RFP documents for extracting requirement statements, and mapping them to offering taxonomies. We also present RFPCog as an interactive and explorative tool for analysis, refinement and browsing of requirements-offering mapping. The presented methods have been applied on RFPs submitted to a large IT service provider company, and the result of evaluation of the methods and tool by practitioners shows the effectiveness of the tool for intelligent requirement extraction and analysis.

1. Introduction

The pursuit of IT Services contracts for large service provides is a very lengthy, costly and people-intensive process. Large client enterprises in sectors such as financial, airlines, manufacturers, etc. go through an IT service procurement process that is often mediated by third party advisors (TPAs) such as Gartner, which on behalf of the client run the procurement process including preparing RFP documents, which are typically large number of documents, ranging from a few to hundreds of documents each consisting of tens or hundreds of pages. These documents describe the business of the client, its current IT environment, its business and IT requirements, and a picture of the future state of the IT environment, and set the bidding rules and guidelines. RFP documents are expressed in natural language text. The current practice for processing RFP documents is manual in which sales team read through these documents to understand the client business, requirements and how they map into current service and product offerings of the enterprise. This is very labor-intensive, tedious, time consuming, error-prone and costly for IT service providers to participate in the bidding process. More and more, IT service providers are under tremendous pressure to prepare RFP to respond within shorter and shorter timeframes.

Due to complexity and diversity of the type and structure of RFP documents and that of natural language that business needs, requirements and asks are expressed in, this problem remains as a challenging problem to this date. In this paper, we focus on the following technical problems from the perspective of an IT service providers:

1) The automated identification of business and IT services requirements from RFP documents. This is a challenging task as it involves digesting textual statements in all RFP documents, as a whole, and understanding which statements, together, constitute a requirement, and what type of requirements including business, functional or non-functional;

2) Identifying the set of provider services that meet the client business and IT requirements. This is non-trivial because clients and providers often do not use the same language and terminology for describing services, and also a mere syntactical matching of the text to services name and description would generate considerable false positive matchings; and

3) Extracting instructions and guiding principles that governs the bidding process including the deadlines, milestones, and a response format
including questions that the client want to be answered in the RFP response.

We present a set of novel linguistic-based methods combined with machine learning for tackling above mentioned problems, and an interactive tool, called RFPCog, which enables continuous learning and improvement of the methods by incorporating users’ feedback on the outputs of the tool. In particular, we make the following unique contributions:

- We present a generic model for a client requirement, defining the key elements of a statement describing a client need, and present a framework and process for automating requirement identification, and requirements-service offering mapping.

- We present a novel method for requirement identification which takes into account linguistic features of statements in RFP documents, and also the context that they are mentioned including the structure of the document, the section, paragraph and adjacent text. This is a staged method starting from requirement candidate identification, and then learns a deeper model for requirement topic identification.

- We present a novel context-aware method for requirement-offering mapping, i.e., identifying the set of provider’s service offerings that meets client requirements, the gaps (requirements that are not met by current offerings), and set of offerings that could be relevant for the customer but are not matched directly to requirements in RFPs. The method considers not only the semantic and partial matching of phrases in requirements and service offering descriptions, but also the context which includes the surrounding text and the structure of RFP documents.

- We present RFPCog, which is an interactive tool that allows a user (a service solution consultant) to explore the results in multiple views, provide feedback which is then taken into account by the tool to improve the learning model that guides the requirement identification, offering matching and extraction for bidding process.

The approaches presented in this paper are generic and can be applied also to contract documents among service providers and clients to identify the agreed-upon services requirements and scope among the sales team, and the client. This is important for the delivery team to identify which services need to be provisioned to clients with what terms. The rest of the paper is structured as follows. In Section 2, we define the problem tackled in the paper. Section 3 describes the requirement extraction, and requirements topic identification model. Section 4 introduces our novel requirement-mapping method. In Section 5, we reviews related work, and we conclude in Section 6.

2. Background and Problem Definition

Service procurement RFP documents describe, in detail, some or all of the followings: the business, its challenges, the goals and objectives for service procurement, the scope of service procurements, current IT environment and services (baselines), the business, functional, and non-functional requirements for assets and services, the desired future state of IT services, etc. Today, a specialized and trained team of business and technical sellers in IT service providers are formed, namely the engagement team, to review the RFP documents, understand the RFP management process, identify service requirements, and the set of service offerings from the IT service providers that meets the client requirements (called solutioning team), based on which an initial price and cost estimation is produced. In this paper, as a first step, we are interested in identifying and grouping statements, expressed anywhere in RFP documents, that constitute a requirement (among all other requirements that do not define a requirement). Let us start by giving an example.

Consider the following statement in a typical RFP document: “Service provider shall provide **onsite Desktop Services** dispatching resources on **24 hour a day, 7 day a week basis**, for **Supported Equipment and Supported Devices** at all **Client’s Service Locations**, which locations may be modified from time to time by **Client** in accordance with the applicable **Change Control Procedure**”. This is rather a well-formed statement of a requirement, in which the following requirement features are observed:

**Responsible Party:** Service Provider  
**Verb phrase:** shall provide

![Figure 1. An example of a requirement statement in form of a table.](image)
**Topic/Service:** Onsite Desktop Services  
**SLA needs:** 24 hour a day, 7 day a week  
**Services for:** Supported Equipment and Devices  
**Locations:** Client’s Service Locations  
**Other Linking Entities:** Change Control

Procedure

However, requirements are not always described in such a well-formed manner: some features can be missing from the statement, the same features may be mentioned in several sentences next to each other or sometimes apart in different documents. They may not be mentioned in text only format either. For instance, Figure 1 shows an example where requirements are expressed in text, as well as in a table, where different features can be found in different columns and rows of the table. There is an additional dimension, where a requirement may have sub-requirements, each providing more detail on a requirement with a larger scope. Identifying such parent-child relationships is also key to the problem of requirement identification.

Next, identifying mapping of identified requirements to the internal service offerings of the provider is non-trivial. Service providers usually maintain a catalog of service offerings which are often in the form of a hierarchy of offerings, called service taxonomy. Nevertheless, the client service needs may not be expressed using the same vocabulary found in that particular service provider’s taxonomy. For this reason, we identify the mapping of a client’s requirements to ITIL (IT Infrastructure Library) directly, and extract client service vocabulary and taxonomies from the RFP documents to provide also a mapping of requirements to those two representation.

### 3. Requirement Identification and Extraction Approach

In this section, we provide an overview of RFPCog. We first start by providing a meta-model for requirements, and then present RFPCog method for adaptive requirement identification and extraction, as well as the mapping to service provider’s service taxonomy.

#### 3.1 Service Requirements Meta-Model

We define a requirement as follows: A requirement is a statement of need, ask or responsibility. A requirement may consist of at least the following features: <responsible party, verb phrase, responsibility topic>, and a number of additional features such as <SLA, Location, Time, Related Entity, Parent>. Requirements in an RFP may be grouped according to their topic to form a Requirement Group. Also, a requirement may be a sub-requirement of another requirement, in which case its parent field will link to the parent requirement. One or more client requirements (the responsible party is “service provider”) may be linked to one or more Service Offering in the provider side (many-to-many relationships). Examples of service offerings are services defined (at various level of granularity) in ITIL service taxonomy, Service Provider’s taxonomy, and Client’s Service taxonomy. The left side of Figure 2 shows high level relationship among modeling elements defined here.

The right side of Figure 2 shows basic elements in RFP documents (elements such as section, paragraph, sentence, figure, table, references, and how they are linked to the requirement model defined above. In particular, a requirement is associated to one or more statements (sentences) in RFP documents. In the following, our method is using information obtained from processing the structure of information to identify which statements (sentences)
are requirements, and which set of them belong to the same requirement.

3.2 Requirement Identification Method: An Overview

The requirement identification in RFPCog framework consists of the following steps, as depicted in Figure 3.

1) RFP Docs Structure Analysis: In this step, the set of RFP documents are analyzed to identify the relationships among different components in the documents starting from the sentence level, to paragraph level, section level (including all text, figure and table in the section), document level, and also cross-document level. At the cross-document level, all sections and text that are cross-referenced are linked together to build a holistic, connected model of all documents in the RFP package. In the training step, the section information is gathered to build a model of section titles that frequently contain requirements, such as “Statement of Work” sections. This information is used as one parameter to assess the likelihood a statement is a requirement.

2) Requirement Candidate Identification: In this step, the first part of speech recognition (POS) tags of statements in the text are identified by applying standard natural language processing (NLP) POS tagger methods. We apply a number of rules that specify how requirements are often expressed by the client, and look for the features that constitute a requirement per the requirement model discussed above. Also, once a requirement candidate is identified, all text following the requirement is considered to be related to the current requirement until the next requirement statement is found. All the text that are related to a given candidate requirement are processed for finding potential sub-requirements.

3) Learning Requirement Identification: In this step, the candidate requirements that are found in the previous step are inspected and processed along with metadata and contextual information to compute a confidence score for each requirement based on the assessment of the values of features found for the candidate requirement, section and context where the statement is located in, the word dependencies and any dependencies among the sentences with the span of a requirement.

4) Requirement Topic Identification: in this step, for each identified candidate, the main topic or the service that the requirement is focusing on is identified. This is important as one of the goals is to find the main services, and related requirements to those services, that are mentioned within the RFP.

3.2 Requirement Candidate Identification
The goal of requirement candidate identification is to prepare an initial set of candidate requirement statements for further, more detailed and deep analysis for requirement analysis through a light and fast process. Consistent with major deep learning methods [9], the requirement identification approach consists of a high level, abstract model for requirement candidate identification consisting of the following elements:

Explicit requirement Pattern: a requirement statement where the subject (responsible party) is explicitly defined. It follows the following structure: <subject> <verb> <object>. The attributes do not need to be in that order in the statement. Examples of most common statements complying with above pattern include: The service provider will supply X, Y, Z. Also, Z, Y, Z has to be supplied by <Name>/provider.

In order to recognize such patterns, we apply linguistic-based rules that represent patterns in the form of <subject: service provider> <auxiliary: will, should> + <requirement action verb: e.g., “is responsible for”, provides, passive action verbs, present tense verbs (whether it is an “ask”).> The above rules operate on part of speech tags and a vocabulary list for the subject and the responsibility verbs. Any synonyms of those keywords and verbs are also allowed under these rules.

Figure 2. High Level Approach for Requirement Information Extraction from RFP Documents
**Implicit Requirement Pattern:** not all parts of the necessary features in the requirement data model may be present in the statement. For example, in many cases subject is not present and the sentence starts with a verb, for instance: “Provide X, Y, Z”. This is the typical case for a bulleted list. The responsibility is determined by the context (generally a reference to the responsible party within the preceding paragraph).

**Structure-based Requirement Patterns:** in some cases, the main features of a requirement may not appear all together and structural clues enable us to identify the start of a requirement. One example is under a given heading, there is no sentence but a list of items, which are assumed as responsibilities of the service provider. The section title may have keywords referring to Responsibilities or in contrary cases to exclusions from the responsibility, which should be understood as well.

To process the content of RFP documents, we focus first on identifying the explicit followed by implicit and structure-based requirements. The same principle applies when processing list item within a list. Tables have their own set of patterns to identify feature values related to requirements. For both lists and tables, the paragraph (or section title, if no text before) that precedes the list or the table is searched for contextual information on the responsibility clues, exclusions or missing feature value (such as a subject). Depending on the feature values found, the requirement candidates in the following list/table are included or excluded in the list of candidates.

### 3.3 Deep Analysis and Learning for Requirement Identification

Figure 3 illustrates the overall process for requirement identification. This step in the process focuses on deep analysis of each candidate requirements to assess the evidences that are found related to each candidate, and make a final decision. The candidate identification step acts as a filter which reduces the space of possible candidate sentences and statements from the RFP documents, which should be examined for requirements. Each evidence is assessed objectively based on an assigned evidence score to indicate the confidence of the information belonging to key features of a requirement, i.e., subject (responsible party), verb (responsibility definition), and object (the topic or service).

**Responsibility Evidence Assessment:** The extracted responsibility evidences are re-evaluated in their context. In particular, the following are among most important supporting evidence:

(i) Signals: various forms of responsibility expressions, deliverable definitions, and dependencies are analyzed. (ii) The following rules are checked on responsibility signals: (1) Signal is head of NP (noun phrase) and SP (service provider)/client is within the NP. (2) Signal is head of NP and SP/client is right before NP. (3) Signal is head of NP, with NP being part of a conjunction and SP/client a conjunct.

The above rules lead to the identification of the following forms of responsibility definition. “following’ + <responsibility signal>, for example in statements such as “provider’s responsibilities include the following”. Also, alternative pattern may be used: <responsibility signal> + ‘following’. For instance, “the following responsibilities being managed and fulfilled by the client”. There are other synonym words that also signify a responsibility definition such as “Upcoming, ‘following’, ‘below’, ‘as follows’, ‘including’, ‘includes’, ‘described as under’, ‘described under’, ‘such as’, ‘include’.

The evidences above all contribute to defining a score between 0-1 which shows the confidence that a responsible party is defined. The guideline for assigning score is that whenever the signals are present within the requirement statement it gets the highest score, and as the signal weakens or mentioned further away from the statement (in terms of number of tokens), the score is reduced proportionally.

The other aspect of a responsibility is identifying whose responsibilities are being defined. In cases, there are explicit mention of “Service provider” or the “Client”, and in other cases neither is mentioned, which we rely on contextual information such signals in preceding sentences, paragraph and the section title. Based on above, a confidence score is defined considering the explicit mention of one of the parties, or implicit one and a lower score is assigned when the signal gets weaker.

**Responsibility Verb Evidence Assessment.** All signal evidences that are collected for each candidate are assessed in order to assign an evidence confidence score. In particular, the goal is to validate the verbs on whether they define valid asks, requests or needs. The main keywords that are looked for (and related synonyms) include: ‘provide’, ‘is responsible for’, ‘obligated’, ‘should offer’, “will perform”, “will supply”, etc. Apart from the vocabulary of the verbs, the type of the verb (whether it is an action verb phrase), the tense of the verb (w.r.t. to the responsible party) and its dependencies are also checked to ensure that the topic/service (below) or the responsible parties are dependent to the given verb. The combination of above signals, whether auxiliary verbs are present and statistical frequency of verbs in
actual requirements (or their synonym similarity) lead to assignment of a score between 0-1 to the responsible verb confidence score.

**Topic evidence assessment.** The third and also an important part of the requirement identification is what the statement talks about, and whether it is focused on one or more of services in any of the service taxonomies in the repository. We explain our method for topic identification in next section, but for the purpose of requirement identification, we find the matching score of the noun phrase that depends on the responsible verb relies (in linguistic term as result of POS dependency analysis of text) [6] a confidence score for the topic is computed. In particular, the similarity of noun phrases in the requirement statement with any service element name in the service provider taxonomy, ITIL service taxonomy, or client service taxonomy is assessed and the highest similarity score (between 0 and 1) is taken as the confidence score of the topic evidences.

The final method for the requirement identification defines a weighted function over these three evidence as follows: $\sum w_i f_i$, $i=1..3$, and $\sum w_i = 1$. Initially, we set the weights for the three aspects to be equal to 0.33. There is an adaptive learning module that takes the feedback of the users in a batch processing mode and adjust each of the weights, adopting a promotion and demotion method as of Winnow algorithm.

**Requirement Score Interpretation and Real-World Field Study:** The above procedure leads into a confidence score between 0 and 1 for each requirement. In the tool, we enable the user to set a configuration parameter to view requirements with Very High confidence score, High Confidence, Average Confidence, or Low confidence requirements (including more of weakly identified requirements). As observed in our human studies, the Very High Confidence, requirements have a high precision and as we get to lower confidence level, the recall (the percentage of requirements identified) increases at the cost of including some non-

requirement statements that follow the same structural and linguistic forms recognized by the approach. Nevertheless, the tool produces high quality results at a High Confidence level score in our field studies with select IT service management RFP and contract documents.

### 3.4 Requirement Topic and Focus Identification

The purpose of topic and focus identification for a given requirement is to understand it talks about which service, and about what aspect of that service. This information enables us to group and present the requirements to the user in an effective manner. Let us start by mentioning that in our context, topic is a noun phrase in a requirements that depends on the responsibility verb, and matches one or more service-related vocabulary in the service catalogs. A given requirement may have one or more topics (noun phrases depending or being associated to the responsibility verb, and qualifying the matching criteria). The focus of a requirement is the topic that is the main purpose of the sentence, which is the main topic associated to the responsibility verb, which also can be known as the “object” of the sentence in linguistic terms. It should be noted that in NLP literature, there are multiple definitions for topic (theme) and focus (comment, rhyme) of a sentence [5], which is specified following a number of rules on
the syntax and semantic of the sentence, and is different but related, to the notion of topic and focus in this paper.

Another difference from the linguistic interpretation of the focus is that the focus of a requirement may be found outside of a given requirement. This is the case for requirements with explicitly missing, and implicitly inferred, topics and focus. Example of places that the missing topic phrases can be found are paragraph containing the requirement, section title, table caption, heading-sentence’s topic of a list.

We treat the problem of topic and specifically the focus identification of a requirement as a machine learning (classification) problem. A number of features are extracted for all noun phrases in the requirement. These features include structural and linguistic features of the requirement to enable us find the structural dependencies between three main parts of a requirement model (subject, verb, and topic). In particular, subject (implicit, or explicit), if subject explicitly present whether it contains a possessive form, the number of noun phrases, the order of the particular noun phrase in the sentence, (the given noun phrase is the Nth noun phrase in the sentence.), the order of the noun phrase as the Mth token in the sentence, whether the noun phrase a part of a predicate, the level in the sentence’s parse tree (= the number of hops from the parse tree’s root node), distance to the dependent responsibility verb, highest Jaccard similarity score for the match with service and provider/client catalog vocabulary including the ITIL taxonomy. As part of features, also we take into account whether the dependency on the responsibility verb is direct or indirect (through intermediates).

As shown in Figure 3, we then train a classifier to identify the main topic (the focus) of the requirement from a list of candidates. In cases the topic and therefore the focus is not present in the body of the requirements, i.e., when the object noun phrase (or any subset of that) does not match any phrase in the service catalogs, we rely on the topics and focus identified in the surrounding context mentioned above. The surrounding context are followed in a hierarchical manner from sub-requirements to requirements, paragraph/table/list, sections, and all RFP documents in search of the focus. In all cases, the topic of other higher level contexts are identified using the same machine learning approach with different feature sets. Note that the matching to service-related concepts and catalogs are of two major types: 1) matching to service element names in one of the service catalogs, e.g. "Mainframe Infrastructure Management", or 2) matching one or more of vocabulary related to service offerings such as “SLA” and “service location”.

4. Requirement to Offering Mapping

Once the requirements are identified, the next step is to identify which requirement is related to which service elements in the offering taxonomies and catalogs. This is highly important for understanding which offerings are within the scope for putting together a solution for the customer. Note that while the topic identification feeds into the offering mapping stage, the topics of a requirement are not necessarily providing a match to services in a given service catalog. The purpose of this section is to introduce more accurate (by introducing a new measure of similarity between requirements and service elements in the catalogs) and pervasive measures to identify the mapping to service offerings in a given service taxonomy and catalog. In the following, we describe our requirement to offering mapping method.

Let NP be a noun phrase in a requirement, and E be a catalog element from any given level of an offering taxonomy. Let us use String_seq to denote the sequence of tokens resulting from tokenize a string (as part of tokenization, all stop words are removed, and words are converted to lower case and the stem of each token is derived). Following this notation, NP_seq represent sequence of a noun phrase, and E_seq denotes the sequence for a catalog element’s name.

In a pre-processing step, catalog taxonomy is analyzed as follows: each element at every level is processed to obtain an E_seq by tokenizing the string, and filtering stop words. Then, each token is converted to lowercase and stemmed. Also, the frequency of each token in the catalog (at any level) is computed.

Novel Method for matching noun phrases in requirements and offerings. In the following, we describe a novel noun phrase matcher, specifically designed requirements-offering mapping, which two sequences (NP_seq and E_seq) from processed terms (corresponding to the NP and E to compare) and returns a similarity score between 0 and 1. The method is a modified Longest Common Sequence (LCS) term matcher. The LCS problem is that of finding the longest subsequence common to all sequences in a set of sequences (two in this case). The main difference of this matcher with other similarity metrics such Cosine and Jaccard is that the LCS preserves the order of tokens in matching, while other don’t. For instance, “project management” and “management project” are equal according to Jaccard and Cosine but for LCS they represent a 50% match.
An observation in the similarity computation of two noun phrases between requirements and offerings is that not all words weigh equally in similarity measure contribution. For example, some words are common to many of the service elements (such as “management”), and therefore frequent, and so not contribute to the same level to identifying similarities between noun phrases as less frequent but unique words such as “asset”. For example, consider two phrases of “asset management”, and “quality management”, while the two have “management” in common, they are not related. Therefore, we define also a weighting mechanism as a co-efficient in identifying the similarity scores between two phrases. In addition, we penalize an LCS score if there are additional or missing tokens in the requirement phrase with the intuition that the additional/missing tokens impact the total matching of the two phrase negatively. This penalization is also done in a weighted manner proportional to the frequency of each word in the catalog.

Let us use SE_Length to denote the length of the E_seq being compared against a noun phrase. SE_Length is used as the denominator in computing the LCS score. This is because the noun phrases can vary in length and contain the exact term among other irrelevant terms. Indeed, we compute a weighted version of the denominator. If LCS is not empty and length of LCS < SE_Length then it means that there are tokens from E_seq not present on NP_seq. If we simply compute LCS/SE_Length the resulting coefficient would be penalizing the missing tokens equally while there are words that are clearly more important than others. For instance: “management” is less important than “mainframe” as the latter is a more specific word in the context. Hence, if a common word is missing in NP_seq, it would be assigned a smaller weight proportional to its frequency. According to this weighting scheme, if the missing term is very specific, the penalization coefficient tends to be close to 1, and if the missing word is very generic, the penalization coefficient tends to be small (closer to 0). We define the base similarity metrics as follows:

\[ \text{Based Similarity Score} = \frac{\#LCS}{\text{Weighted Denominator}} \]

where Weighted_Denominator is defined as the weighted sum of the number of missing words in the E Seq.

In addition to considering the length of the two noun phrases and whether there are additional/missing terms, it also matters if the matching terms are mentioned close-by or there are terms in between that discount the quality of the match. To take this into account, we also introduce a penalization for the distance between the matching terms. The objective is to penalize NP_Seq whose matching tokens are too far apart with respect to the E_Seq. For instance, for the following examples the base score would be 1, though Case B clearly refers to a different scope and should be reflected:

A) <NP: “Storage management”, E: “Storage Management”>. And,

B) <NP: “storage systems / asset management”, E: “Storage Management”>

Let us define NetDistance be the absolute difference in tokens between the LCS in NP_Seq and E_Seq. For the previous two examples, the NetDistance for (A) is 0 and NetDistance for (B) is 2. In another example, of E: “Storage XYZ ABC Management” and the same NP: “storage systems / asset management”, and the LCS <Storage Management>, the resulting NetDistance is 0 as the two matching tokens are equally far apart in both cases. In this example there is no extra penalization due to distance but the score will be lower due to only two out of four tokens matched on E. Given Above, we now define the final

\[ \text{Final Similarity Score} = \text{Based Similarity Score} \times (1 - \text{net_distance}/C) \]

in which C is a constant for the maximum length of noun phrases in the population.

Through the use of a configure threshold, the list of candidate matching is filtered to the ones with a similarity score above the threshold.

5. Implementation and Case Study

We have implemented a proof of concept for the presented method and conducted experiments to evaluate the effectiveness of two aspects of the tool, i.e., its requirements and topic identification, and the requirement-offering mapping and explorative analysis and user feedback loop.

Implementation. The backend of the tool is developed in Java. For NLP tagging and annotation, we use SystemT information extraction methods and library from IBM, which is part of IBM BigInsight products [1]. The front-end of the tool is developed in Javascript, and in particular with AngularJS and D3 library for the charts and visualization components.

Requirement and Topic Identification Evaluation. Figure 4 shows the result of the evaluation of the topic identification method and comparing three main classification method (Naive Bayes, Logistic Regression and Support Vector Machines (SVM)) using the features the were explained in Section 3.4. In this experiment, we had more than 300 requirements, for which all their
features were extracted and the main topic (focus) were identified by human experts.

The reported results are based on 10-fold cross validation, in each round noun phrases from 30+ requirements were used as testing while the rest of the data were used for training. As shown, the derived set of features achieves a high precision in the identification of the main topic (focus) of each requirement. And, in particular, among the three classifier, SVM achieves the highest precision, which we attribute it to the complex non-linear model that is needed to distinguish topics from non-topic noun phrases in requirement statements. And, the table below the figure shows the details of SVM performance on different metrics including true positive, false positive rate, precision, recall, F-Measure for each of True and False classes.

**Interactive and Explorative Requirement-Mapping Evaluation.** We developed a graphical user interface which evaluated it with a number of expert users. The tool provide two complementary views (Figure 5). One is focused on Requirements, which shows the text of a given RFP or contract document that is highlighted and annotated with requirement information. This view is helpful as the user can see the annotated text formed into requirements, and also identified mappings to service elements in a specific service taxonomy and provide in-context feedback on each requirements and mapping. The other view (showed in Figure 5) shows a visualization of the grouping of the requirements based on the mapping of requirements to offerings. The offerings with one or more matching requirements are shown in a layered circular chart (aka sunburst chart) according to their level in the taxonomy of the given catalog. The tool supports three different catalogs: Service Provider, ITIL and Customer Service Catalog (extracted by our method).

Clicking on each segment of the chart results to filtering the set of shown requirements in the view below the chart to the corresponding matched requirements. The table below shows the reason (piece of text) that resulted in the match. There is also mechanism for providing feedback by the user on the identified requirement mapping to the service. This view provide three complementary perspective for each offering catalog: showing service elements with a matching requirement (default), showing requirements without a service offering match (to identify gaps in the offering space and potential partnering), and related offerings without a strong service requirement match (opportunities to cross sell related services to the customer).

The result a user study with experts showed that the tool achieves a high precision in identifying the requirements in RFP and contract documents. We had a select number of pilot users to use this tool and compare the results achieved through their team manually reading RFP and contract documents. They reported that the tool was comprehensive in identifying a large portion of the requirements (still missed almost 10% expressed in non-conforming forms). In some cases also the offering mappings, though syntactically correct were not semantically a mapping to the right level of service granularity. Nevertheless, the tool was deemed effective in identifying a number of requirements that the team has missed.

**Figure 4.** The evaluation of topic identification method. Legends: TP (True Positive), FP (False Positive), ROC (Area under the Curve)

![Figure 4. The evaluation of topic identification method.](image)

**Figure 5.** The screenshot of RFPCog tool for requirement extraction and mapping requirements to service taxonomies.

![Figure 5. The screenshot of RFPCog tool for requirement extraction and mapping requirements to service taxonomies.](image)
6. Related Work

The problem of information extraction from text has been studied extensively [6, 7] from various sources including documents, Web and in different domain. These approaches offer basic tools and methods for annotating text. A second class of work in this space [7] focuses on defining and learning patterns and text structures from a given source such as a Web page. The problem of automatic extraction of requirements from textual software descriptions has received interest in the context of software engineering [2] and in system engineering domains [3]. In [2] authors present a method for transforming the textual software requirements to SBVR (Semantic Business Vocabulary and Rules) requirements specification. It focuses on extracting different concept, entity and object types defined in SBVR model and rules from the source text.

The authors in [3] investigate the problem of processing textual descriptions of software systems to extract information according to concepts, entities and type sin Object-Process Methodology (OPM) model. The approach in this paper is similar to that of [2] in the sense that the information extraction is guided by a meta-model with certain object types and models. As another related work, authors of ARSENEL [4] focus on extracting information from textual descriptions related to software requirements. The problem that is parsing the text to find the details of the formal specification of a software expressed in LTL (linear-time temporal logic). It starts by building the dependency graph of certain entities of interest, and uses it to build a predicate graph, which feeds building logical formulas.

This paper is the first to investigate the problem of requirement extraction from natural text from RFP documents, and specifically those from services domain. In contrast with above referenced work, which their process of information extraction is guided by a specific model such as SVBR, OPM, or entity dependencies, our requirement extraction model is very generic in principle. It introduces a staged, deep learning based method, which learns from user feedback and benefit from evidence-based scoring mechanism, similar to the deep learning method introduced in IBM Watson system in Jeopardy game [9]. In addition, we introduce a novel requirement to offering matching algorithm that considers the context of the match in a weighted manner.

7. Conclusion and Future Work

In this paper, we introduce a set of novel methods, and RFPCog tool for linguistic-based identification of service requirements expressed in natural language in RFP and contract documents in IT services domain. To the best of our knowledge, this is the first method for requirement extraction from RFPs, and in services domain. We described the design and development of a novel framework to automatically extract customers’ requirements, and identify their topics (service focus) using features extracted based on NLP and contextual information in the RFP documents and applying machine learning techniques for topic identification, and for adaptive and multi-layer (deep) learning of a model of the requirements identification model. We also presented the result of quantitative and qualitative evaluation of the methods and RFPCog tool.

As a future direction, we are envisioning to integrate a question and answering system on top of the extracted requirement knowledge graph to enable answering questions to sellers about the client’s requirements. Also, another interesting question is the detection of contradictory requirements, and details about servers, locations, etc. that are provided in RFPs and link them back to the corresponding requirements.

10. References