Automatic Process Model Discovery from Textual Methodologies

Elena Viorica Epure, Patricia Martín-Rodilla, Charlotte Hug, Rebecca Deneckère, Camille Salinesi

Automatic Process Model Discovery from Textual Methodologies

An Archaeology Case Study

Elena Viorica Epure[1], Patricia Martín-Rodilla[2], Charlotte Hug[1], Rebecca Deneckère[1], Camille Salinesi[1]

[1] Université Paris 1 Panthéon-Sorbonne
Centre de Recherche en Informatique
Paris, France
Elena.Epure@malix.univ-paris1.fr

[2] Institute of Heritage Sciences
Spanish National Research Council
Santiago de Compostela, Spain
patricia.martin-rodilla@incipit.csic.es

***Abstract*— Process mining has been successfully used in automatic knowledge discovery and in providing guidance or support. The known process mining approaches rely on processes being executed with the help of information systems thus enabling the automatic capture of process traces as event logs. However, there are many other fields such as Humanities, Social Sciences and Medicine where workers follow processes and log their execution manually in textual forms instead. The problem we tackle in this paper is mining process instance models from unstructured, text-based process traces. Using natural language processing with a focus on the verb semantics, we created a novel unsupervised technique *TextProcessMiner* that discovers process instance models in two steps: 1.*ActivityMiner* mines the process activities; 2.*ActivityRelationshipMiner* mines the sequence, parallelism and mutual exclusion relationships between activities. We employed technical action research through which we validated and preliminarily evaluated our proposed technique in an Archaeology case. The results are very satisfactory with 88% correctly discovered activities in the log and a process instance model that adequately reflected the original process. Moreover, the technique we created emerged as domain independent.**

Keywords—process mining, process mining technique, natural language processing, process model, technical action research

I. INTRODUCTION

Although Humanities have increasingly adopted Digital Research, there is still a significant resistance to changing the traditional research methods mostly based on the manual production of textual sources and qualitative information [1, 2]. This issue is also fed by a misalignment of the existing information systems to adequately manage, analyse and operate the specific type of information resulting from the Humanities research [3, 4, 5, 6].

Contrary, other areas such as business have fostered a plethora of research in this respect leading to the actual emergence of several standalone disciplines such as knowledge management [7], method engineering [8] and process mining [9] together with their various solutions. We focus in particular on process mining driven by the fact that the Humanities workers keep a record of the processes they follow as textual sources. We believe that by exploiting these textual sources for mining process models we could facilitate and improve the teamwork and the knowledge sharing. Moreover, we could enable a common ground for the comparison, validation and centralisation of the applied processes and methodologies. These are already proved benefits of process mining [9].

We then decided to focus on Archaeology in the beginning, given an established collaboration of the authors with this community. We consider the methodology section of the archaeological report a text-based process trace where the archaeologist describes the process the team followed during their work. This section might appear under slightly different titles -methodology, survey methodology, excavation methodology, evaluation methodology- but it has the same goal. Example of archaeological reports can be found at [10], the British public repository. Thus, the research question emerging from these practices we aim to answer is: How to use textual methodologies for producing structured knowledge as process models?

Before presenting our solution, we want to clarify several theoretical concepts: activity, process, process instance, process model and process mining. An *activity* is a task, which once completed leads to a full or partial accomplishment of the goal it is related to. The activity is the building block of a process. A *process* consists of a collection of activities and their ordering [9]. The ordering of the activities is represented by several activity relationships: sequence, parallelism and mutual exclusion [9]. More complex relationships exist but we currently consider only these. The mutual exclusion implies in fact a decision in the process flow: an activity is chosen over others based on some decision parameters. Consequently, a process could be executed in multiple ways. The *process instances* are different executions of the same process. Further, a *process model* is the representation of a process. Finally, *process mining* is the discipline aiming at automatically discovering process models from process traces [9]. Most of the work in process mining so far has considered that the process traces are the logs extracted from the information systems where the processes were executed [9, 11, 12]. However, there are other areas such as Humanities where processes exist and are followed without a similar support from the information systems.

We envision a solution in two steps: (1) mine process instance models; (2) mine process models by aggregating process instance models. In this paper we focus on step (1) while step (2) is left as future work. The technique we created, *TextProcessMiner,* generates the log of activities from text and then discovers the process instance model. Compared to other related works [13, 14, 15, 16, 17], *TextProcessMiner* is fully unsupervised and uses natural language processing techniques with a focus on the verb semantics. The technique has emerged as a result of technical action research [19]. We validated it in a specific case in Archaeology though it can be applied to any other domain where processes are described in natural language. The results we obtained during validation are very satisfactory with 88% correctly discovered activities in the log and a process instance model that adequately reflected the original process.

The paper is organized as follows: Section II describes the followed methodology, Section III introduces a demonstrating example of the proposed technique, Section IV presents the technique in details, Section V presents the evaluation, Section VI discusses the related work and Section VII highlights the conclusions and perspectives.

II. Methodology

The followed research methodology is Technical Action Research (TAR), an approach centred on technique creation [19]. The goals are to design a technique, use the technique to help the identified stakeholders – in our case humanities specialists- and reflect on the benefits and limitations of the technique in a practical case. Technical Action approach emerged naturally as the most appropriate research methodology for us, for two main reasons. First, all the authors of this research work are affiliated with humanities institutions. This situation allows us to address this research not only in terms of problem identification, but also in terms of actively seeking technological solutions for improving the situation in the real context. This study of artefacts in context, not only as diagnosis but also as intervention, is by definition TAR [19]. Second, Technical Action Research is technology-driven, being employed for learning more about the created technique which might after be used for solving various problems, similar in architecture with the case where it was initially validated [19]. This characteristic of TAR provides more flexibility to study the implications in the real context of the created technique than in other research methodologies as case study [20] or design science [21].

In the next section we discuss further the three phases of Technical Action Research: problem investigation, technique design and technique's validation.

A. Problem Investigation

In this first phase, we have identified the problem our technique might deal with: process-mining solutions are not currently adapted for exploiting the unstructured information created by the Humanities workers.

Archaeology is a suitable application domain for our approach given the following reasons:

- Archaeological practices, as a humanities discipline, generate a large amount of textual sources;
- There is an increasing demand of textual analysis solutions for supporting the humanities;
- The methodology sections from the archaeological reports describe processes in natural language.

B. Technique's Design

We carried out an initial screening of the textual corpus in order to identify the technique's objectives: (1) clean the methodology section, (2) discover the process activities and (3) discover the process instance.

This phase implied the creation of a potential design of the solution for the identified problem. The design was brainstormed separately for each technique's objective and afterwards integrated to yield the final version.

The development of the technique followed the design we defined in the previous phase. The partial testing results outlined improvements areas, triggering changes in the design. This also reflected in multiple iterations in the development process.

C. Technique's Validation

We validated the proposed technique in an archaeological case that allowed us to gather detailed observations. Moreover, the implied in-depth analysis helped us to identify the solution's strengths and weaknesses together with potential perspectives.

The selected report consisted of the description of the archaeological works carried out in the site of Villa Magna [22, 23]. The site lies in the Valle del Sacco in Lazio, south of the town of Anagni (Italy). The final report with the methodology section we used was published in 2010 [22].

In order to perform a rigorous validation, the archaeological report used in the case study was different than the corpus we used for testing the solution. Moreover, we defined a protocol with two questionnaires regarding the validation of the logs and the validation of the models. In addition, we created a third group of questions for a preliminary evaluation to discover the solution' strengths and weaknesses from the point of view of the potential users.

III. Illustration

In this section, we demonstrate the technique using a fragment extracted from the methodology section of the report "Land at Station Road Honeybourne Worcestershire Archaeological Excavation"[1]. In this case for example, the methodology section describes the excavation and all other related activities, such as the analysis of the archaeological

[1] http://archaeologydataservice.ac.uk/archives/view/cotswold2_WSM49638/

finds, their recording and documentation. Consequently, a methodology section is a process trace capturing the activities completed during the process and their ordering.

We noticed while analysing manually the archaeological reports that sometimes the methodology section contained a description of the same process (the excavation), but for different areas. Normally, the text referring to each process instance (the excavation of each separate area) should be identified. Each process instance should be mined separately and all the resulted process instance models should be aggregated for obtaining the final process model. While we recognize this situation and we identified the requirements of the final more general solution, we decided to select from the archaeological reports only those whose methodologies contain a single process instance, leaving the other cases for future work. In TABLE I a fragment of the archaeological methodology is presented before and after the cleaning.

TABLE I. EXAMPLE OF A METHODOLOGY FRAGMENT – BEFORE AND AFTER CLEANING

1.14 The archaeological works comprised the mechanical removal of non-archaeologically significant soils, under constant archaeological supervision, using a toothless ditching bucket. The machining ceased when the natural substrate was revealed. All archaeological features were recorded in plan using a Leica 1200 series SmartRover GPS and surveyed in accordance with CA Technical Manual 4 Survey Manual (2012). 1.15 (…) no deposits were identified that required sampling. (…)
The archaeological works comprised the mechanical removal of non_archaeologically significant soils , under constant archaeological supervision , using a toothless ditching bucket . The machining ceased when the natural substrate was revealed . All archaeological features were recorded in plan using a Leica 1200 series SmartRover GPS and surveyed in accordance with CA Technical Manual 4 Survey Manual .

Initially, the text is processed and cleaned. Specifically, the following actions are taken:

- Remove "1.14" and "1.15", the numbering preceding each paragraph;
- Replace "-" by "_" in "non-archaeologically";
- Write each sentence in a separate line;
- Prefix all the punctuation signs with space;
- Remove the text in the parentheses "(2012)" because it does not contain any verb;
- Remove the sentence "no deposits were identified that required sampling" because it contains the negation in the form of "no" + noun. However, keep the sentence with "non-archaeologically" because the negation in the form of 'non' + adverb does not induce a negation in the semantics of the whole sentence.

Further, the objective is to discover the log of activities. During the initial screening of the methodology sections, we noticed the activities were introduced most of the time by verbs and in some few cases by nouns derived from the corresponding verbs (e.g. "the removal of spoil"). An activity by definition implies someone doing something[2]. Consequently two parts compose an activity: the verb and its object(s). The propriety of the verb taking objects is called transitivity[3].

TABLE II. TREEBANKS FOR THE FRAGMENT\ SENTENCES

<table>
<tr><th>Sentence 1</th><th>Sentence 3</th></tr>
<tr><td>(ROOT
(S
(NP (DT The) (JJ archaeological) (NNS works))
(VP
(VBD comprised)
(NP
(NP (DT the) (JJ mechanical) (NN removal))
(PP
(IN of)
(NP
(ADJP (RB nonarchaeologically) (JJ significant))
(NNS soils))))
(, ,)
(PP
(IN under)
(NP (JJ constant) (JJ archaeological) (NN supervision)))
(, ,)
(S
(VP
(VBG using)
(NP (DT a) (NN toothless) (VBG ditching) (NN bucket)))))
(. .)))</td><td>(ROOT
(S
(NP (DT All) (JJ archaeological) (NNS features))
(VP (VBD were)
(VP
(VP (VBN recorded)
(PP (IN in)
(NP (NN plan)))
(S
(VP (VBG using)
(NP (DT a) (NNP Leica) (NNP 1200) (NN series) (NNP SmartRover) (NNP GPS)))))
(CC and)
(VP (VBN surveyed)
(PP (IN in)
(NP (NN accordance)))
(PP (IN with)
(NP (NNP CA) (NNP Technical) (NNP Manual) (CD 4) (NNP Survey) (NNP Manual))))))
(. .)))</td></tr>
<tr><th colspan="2">Sentence 2</th></tr>
<tr><td colspan="2">(ROOT
(S
(NP (DT The) (NN machining))
(VP
(VBD ceased)
(SBAR
(WHADVP (WRB when))
(S
(NP (DT the) (JJ natural) (NN substrate))
(VP (VBD was) (VP (VBN revealed))))))
(. .)))</td></tr>
</table>

Existing works in natural language processing support the identification of the verb and its objects in the text: the parsers. A natural language parser is able to grammatically analyse the structure of a sentence and produce a treebank. A treebank is a representation of a sentence as a tree with annotations at different levels: clause level, phrase level and word level [24]. At the clause level, for example, the speech tag S marks a simple declarative clause while the tag SBAR marks a clause introduced by a subordinating conjunction. At the phrase level, the most interesting for our objective are NP - which marks a noun phrase, VP - which marks a verb phrase and ADJP – which marks an adjective phrase.

[2] http://www.oxforddictionaries.com/definition/english/activity
[3] http://en.wikipedia.org/wiki/Transitive_verb

Finally, at the word level, each word is marked with a speech tag: NN for noun singular, NNS for noun plural, VB for verb, VBG for verb in "-ing" form, VBN for verb in past participle form etc. A complete list of all the tags can be found at [25]. The treebanks are presented in TABLE II.

All the VP and ADJP sub-trees are checked if they contain at least one transitive verb[2]. To identify the verbs we extract the leaves' values of the VB, VBD, VBN and VBG sub-trees. The sub-trees containing the verbs are highlighted in TABLE II too. The values "was" and "were" are not considered because the verb "be", though transitive, is not an action verb and, in this situation, it has an auxiliary role in the passive voice.

TABLE III. ACTIVITIES EXTRACTED FROM THE FRAGMENT

Sentence 1	Sentence 2	Sentence 3
1.comprise removal_of_soil 2.use toothless_bucket	3.cease machining 4.reveal substrate	5.record feature 6.use leica_series_smartover_gps 7.survey feature

Next, we discover the objects for each verb by checking if the sub-tree introducing the verb has, as right sibling, a NP sub-tree. This is the case for: Sentence 1 - "(VBD comprised)", "(VBG using)"; Sentence 3 - "(VBG using)". For all the others, it is checked if their corresponding VP or ADJP sub-tree or any of their ancestors have a NP sub-tree as left sibling. The first NP clause found is taken. Additionally, if a node of type S is encountered the search stops. This is the case for: Sentence 2 - "(VBD ceased)", "(VBN revealed)"; Sentence 3 - "(VBN recorded)", "(VBN surveyed)". For "(VBG ditching)" in Sentence 1 no objects are found following the defined method, therefore it is not considered an activity. The verbs and the nouns in the NP elements are extracted and lemmatized, obtaining the activities in TABLE III.

TABLE IV. SYMBOLS IN PROCESS MODEL REPRESENTATION

Symbol	Usage	Example
_	For activity names: when the object is composed of multiple nouns or when the verb has a particle	area_of_trench carry_out
->	For activity relationship: when an activity follows another activity (*sequence*)	excavate trench -> inspect soil
||	For activity relationship: when two or more activities are executed in the same or overlapping time (*parallelism*)	excavate trench || collect find
x	For activity relationship: when there is a decision between two or more activities (*mutual exclusion*)	take photograph x draw plan
()	For activity relationship: when influencing the precedence of the relationships	(excavate trench || collect find) -> draw plan

Finally, we discover the relationships between the activities by applying a set of rules from a knowledge base we define. The special symbols we use for representing these relationships and the final process are explained in TABLE IV.

Prior, the sentences are transformed: (1) by replacing the verbs from the activities with their tags; (2) by keeping key structures as prepositions, conjunctions, punctuations; and (3) by replacing everything else with the placeholder "…". The results of the pre-processing are presented in the first line of each sentence in TABLE V. The rules we apply are presented in the second line and the result after applying the rules are in the third line, for each sentence. The final process instance is presented in the last line.

TABLE V. PROCESS INSTANCE DISCOVERED FROM THE FRAGMENT

Sentence 1	(S… 1.VBD … , … , (S 2.VBG … .))
	"1.VB/VBD/VBN … 2.VBG " => 1||2
	(1||2)
Sentence 2	(S… 3.VBD when (S … 4.VBN .))
	"1.VB/VBD/VBD … when… 2.VB/VBD/VBN" => 2->1
	4->3
Sentence 3	(S … 5.VBN … (S 6.VBG …) and 7.VBN … .)
	"1.VB/VBD/VBN … 2.VBG " => 1||2 "1.VB/VBD/VBN/VBG … and … 2.VB/VBD/VBN/VBG" => 1->2
	(5||6) -> 7
(1.comprise removal_of_soil || 2.use toothless_bucket) -> 4.reveal substrate -> 3.cease machining -> (5.record feature || 6.use leica_series_smartover_gps) -> 7.survey feature	

When we find the pattern ", … ," between two verbs in the text, we consider it as an explanation, addition or detail. This is the reason why we replace it with "…" in Sentence1 before applying the actual rule.

The parentheses and the tag S are ignored when applying the rules. We, nonetheless, keep them as they influence the grouping of activities. In Sentence1 and Sentence 3 the activities which are extracted from dependent clauses will be grouped: (1||2) and (5||6). Moreover, the parentheses and the tag S allow us to decide the pair of verbs for which rules are checked. The default pairing takes two consecutive verbs as they appear in the sentence. In Sentence 3, the second rule is applied to 5.VBN and 7.VBN instead because of the reasoning over the clauses. For this illustration, we do not have any verb with multiple objects. In that case, the activity would have been broken in multiple parallel or mutual exclusive activities, which are also grouped.

IV. SOLUTION

In this section, we introduce *TextProcessMiner* (Fig. 1) the technique we created for mining processes from text. Although there are similar works discussed in more details in Section VI, our approach is fully unsupervised and uses natural language processing techniques with a focus on the verb semantics. By understanding the verb semantics: we handle the passive/active sentences implicitly and we minimize the number of false positive activities enforced by the discovery of only transitive verbs.

Further, we discuss in depth the three components of the whole solution: *TextCleaner* - responsible for cleaning the methodology section and preparing the text which will be mined; *ActivityMiner* - responsible for mining the activities from the text; and *ActivityRelationshipMiner* - responsible for mining the relationships between the activities, thus discovering the process instance. Currently, *TextCleaner* is part of the whole solution but not part of the mining technique because it does not handle the variability of different textual sources. Contrary, given any cleaned text about a process, *TextProcessMiner* could handle it.

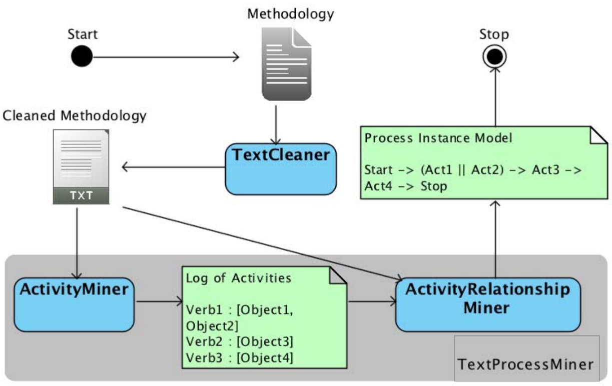

Fig. 1. Solution Overview

We implemented our technique, following the technical action research principles, focusing on delivering fast results and thus enabling early experiments. The results of these experiments were further used for improving the tool output during multiple iterations. In the final experiments, the authors, as process modeling experts, were also involved to evaluate the results and provide feedback. The tool was developed in python [26] using different natural language processing libraries: NLTK [27], Stanford [28] and Enchant [29].

A. *TextCleaner*

The input of *TextCleaner* is the report's methodology. Initially, the processing of the text consisted in the section extraction and sentence segmentation. However, after multiple experiments with different reports and the analysis of the results we decided to introduce cleaning actions to improve the output. These actions are summarized further:

- Replace "-" with "_";
- Add a space before and after each punctuation sign;
- Remove "'" from the possessive singular nouns;
- Remove the comments introduced by parentheses if no verb was found inside these comments.

We also removed the sentences that contained negations because we assumed the activities were introduced by positive structures: they represent something that was done. After analysing our corpus, we noticed negations can be linked to a noun: "no" + noun, "not" + noun, "none of" + noun, or can appear in relationship to a verb: "not" + verb, "-n't" + verb, "cannot" + verb. We also identified several exceptions, when a negation keyword might appear in an expression that is not a negation (e.g. "not only … but also"). In this situation, we do not discard the sentence.

The output of *TextCleaner* is a text file containing one sentence per line. This text file is further fed into *TextProcessMiner* as input and used by its sub-components: *ActivityMiner* and *ActivityRelationshipMiner*.

B. *TextProcessMiner*

TextProcessMiner is a process mining technique because it discovers the process instance models automatically. However, it distinguishes from other process mining techniques by having as input textual process traces and not system-captured logs. Discovering the process in this situation becomes a matter of text mining.

The partial output of *TextProcessMiner* is a log text file with all the discovered activities. The final output is a text file with the process instance model (see Fig. 1).

1) *ActivityMiner*

The goal of *ActivityMiner* is to mine activities. We formulate the following requirements of *ActivityMiner*:

- The algorithm should identify *the transitive verbs* which are most probably linked to an activity;
- For each discovered transitive verb, the algorithm should identify *its object(s)*.

First, *ActivityMiner* produces the treebank for each sentence using the Stanford and NLTK taggers, and the Stanford parser. We opted for the Stanford parser for its very high accuracy [28] and its online version used to test our results. Though the Stanford parser comes as a Java library, NLTK contains an interface[4], which allowed us to continue working in Python.

The Stanford parser is able to produce a treebank either directly from a raw text or from a tagged sentence - a list of tuples containing the sentence's words and their part of speech tags. Initially, the parser was used in the first manner, creating treebanks from raw texts. However, later we decided to explore both possibilities as relevant words were wrongly tagged sometimes (nouns being marked as verbs, verbs being marked as adjectives etc.). We also noticed the speech tagging could be perform either by using the Stanford POS Tagger [30] or NLTK POS Tagger [27]. An experiment was set up to measure which possibility of the following yielded the most accurate and complete results when parsing five archaeological reports:

- Using the Stanford parser with raw text;
- Using the Stanford parser with the text prior tagged with the Stanford tagger;
- Using the Stanford parser with the text prior tagged with the NLTK tagger.

[4] http://www.nltk.org/_modules/nltk/parse/stanford.html

The best results were obtained with the second option. However, sometimes the NLTK tagger seemed to be able to correct some of the errors of the Stanford tagger. Therefore, we decided to use the result of the Stanford tagger as the base input of the parser but to artificially and automatically modify it by favouring the NLTK tags in the following situations:

- If the NLTK tagger tags the word as noun (NN, NNS) and if the Stanford tagger tags the word differently but not as verb in the "-ing" form (VBG) followed by a preposition (IN), and not as verb in the past form (VBD), and not as a verb in past participle form (VBN);
- If the NLTK tagger tags the word as verb in the base form (VB) and if the Stanford tagger tags the word differently but not as verb in any other form (VBG, VBD, VBN) and not as noun (NN, NNS).

Having found the method to obtain the most accurate treebank, we focus then on identifying the transitive verbs. Identifying the verbs in a sentence is a straightforward task: we look in the treebank for the VP or ADJP sub-trees to identify its children who start with VB, if any. However, for the automatic classification of verbs as transitive and intransitive extra-knowledge is needed. Two sources - VerbNet [31] and WordNet [32] - were used for compiling a dictionary of verbs having as key the verb in the infinitive form and as value a Boolean flag for the transitivity. Though the sources are different, they both share common information about the verbs: the frames. A frame illustrates how a verb could be used in a simple sentence [31]. The transitivity in VerbNet is considered true if the frame "NP V NP" is found among the verb's frames and if the first NP has the semantic role of Agent [31]. An agent is an active, intentional entity that carries out the activity introduced by the verb [31]. With the later condition some of the transitive verbs, which are rather states than actions, were excluded. Likewise, we parsed the WordNet [32] corpus and we added more verbs to the dictionary. In this case, the transitivity was judged depending on whether the frame "Somebody verb Something" was among the verb's frames. It can be noticed that "Somebody" is equivalent to a NP element with the semantic role of Agent. Two verbs were artificially changed to being intransitive: "be" and "have", because they do not represent activities. Moreover, during the experiments, when transitive verbs that were not in the dictionary were discovered, we added them manually to our file storing the verbs dictionary.

The verbs can be in active or passive forms. Consequently, the object of a verb can appear as object or subject. The form of the verb is not checked. Instead, we first check if the verb has an object. In the case it does not, we consider the subject being the verb's object. This is in general a valid assumption as we work with transitive verbs and passive voice is often used in reporting. For finding the object after the verb, we search for a NP element being the first right sibling of the VB sub-tree. For finding the object before the verb we look for the first left sibling of the VP or ADJP sub-tree or of any of their ancestors. A verb can have multiple objects in an enumeration or an object composed of multiple nouns. The algorithm discovers both cases.

Algorithm 1 Function to mine activities from a sentence

Require: $sent$ the input sentence and $Verbs$ the dictionary of verbs
Ensure: $Activities$ the list of tuples ($verbs$, $objects$)

```
function MINE_ACTIVITIES(sent, Verbs)
    Activities ← []
    tagged ← TAG_SENTENCE(sent)
    treebank ← PARSE_TAGGED(tagged)
    for all subtree ∈ subtrees(treebank) with node label VP or ADJP do
        verbs ← FIND_VERBS(subtree, Verbs)
        if #verbs = 0 then continue
        end if
        objects ← FIND_OBJS_AFTER_VB(subtree)
        if #objects = 0 then
            objects ← FIND_OBJS_BEFORE_VB(subtree)
            if #objects = 0 then continue
            end if
        end if
        activities ← FORM_ACTIVITIES(sent, verbs, objects)
        extend Activities with the newly discovered activities
    end for
    return Activities
end function
```

The result of *ActivityMiner* is a list of tuples where the first element of the tuple is the verb and the second element of the tuple is the list of objects. Both verbs and nouns are lemmatized using the WordNet lemmatizer [27]. The pseudo-code for the main algorithm of mining the activities from a sentence is summarized in Algorithm 1.

2) ActivityRelationshipMiner

The goal of *ActivityRelationshipMiner* is to mine the relationships between the discovered activities. As mentioned before, there are three types of relationship: sequence, parallelism and mutual exclusion.

In the archaeological report's methodology section the writer reports the events in the order they happened. Therefore, the default relationship between two consecutive activities is sequence. Additionally, we also consider that two sentences are sequential by default. However, if the order is different then there are clues in the text which announce the change: for example temporal structures as "last year", "in 2011" or prepositions / conjunctions as "before", "after", "and", "or", "in order to".

We propose *ActivityRelationshipMiner* as a rule-based system. The knowledge base represents a set of rules defined after analysing the corpus. Some rules are presented in TABLE VI. The activities extracted by *ActivityMiner* for each sentence together with the sentence are fed to the algorithm. First, the sentence is transformed by keeping only the tags of the verbs found among activities, the S tags and their corresponding parentheses and other key structures as conjunctions, punctuation, prepositions etc. All the other words are replaced with "…". We show two examples:

- The sentence "The excavations were structured to accommodate the requirements of the developer." becomes "… 1.VBN to 2.VB … ." ;
- The sentence "Particular areas were targeted by the second machining, including sondages across ditches

and enclosure interiors." becomes "… 1.VBN … , 2.VBG … and … .".

The numbers are used to relate the transitive verbs to the discovered activities.

TABLE VI. RULES TO MINE ACTIVITY RELATIONSHIPS

Rule Input	Rule Input
… 1.VB/VBN/VBD … , … 2.VBG …	1 -> 2 (independent clause)
… 1.VB/VBN/VBD 2.VBG …	1 || 2 (dependent clause)
… where VB/VBD/VBN … 2.VB/VBD/VBD …	1? -> 2 (decision)
… 1.VB/VBN/VBD … , 2.VB/VBN/VBD … or 3.VB/VBN/VBD …	1 x 2 x 3 (branches)
… in order to 1.VB … , … 2.VB/VBN/VBD …	2 -> 1 (sequence)

The rules are always applied for a pair of verbs. For extracting the pairs, we take in consideration the sentence's clauses and the verbs dependencies to each other. After discovering the relationship for each pair of verbs, the process fragment for the complete sentence is composed. Finally, the process fragments obtained from all sentences are put together to obtain the process instance. Currently, we consider the process fragments are in sequence but, in the future, we want to order the sentences and the independent clauses based on time structures. There is already research in this direction, one example being Stanford Temporal Tagger [33] or the work of Muller [34].

We also consider the situation of the verbs having multiple objects. The activity is transformed in multiple parallel activities, one for each object, if the objects are enumerated with "and". The activity is transformed in multiple mutual exclusive activities, one for each object, if the objects are enumerated with "or". Finally, the algorithm is summarized further:

Algorithm 2 Function to mine the process instance fragment for a sentence

Require: *sent* the input sentence and *Activities* its list of activities

Ensure: *ProcessFragment* a string with the activities and their relationships represented with the symbols from Table I

function MINE_PROCESS_FRAGMENT(*sent*, *Activities*)
  transformed ← TRANSFORM_SENT(*sent*, *Activities*)
  pairs ← EXTRACT_VB_PAIRS(*transformed*)
  relationships ← []
  for all *pair* = (*v*1, *v*2) ∈ *pairs* **do**
    r ← APPLY_RULES(*pair*, *transformed*)
    add *r* to *relationships*
  end for
  ProcessFragment ← BUILD_STRING(*relationships*, *transformed*)
  return *ProcessFragment*
end function

C. *Discussion*

Though the presented technique achieved very good results on the selected corpus and during the validation, there are parts that could be improved.

First, *TextCleaner* removes the whole sentence if one of the negation structures is found. A better selection should be made, as there are situations when a complex sentence with negation contained also activities. One solution would be to remove only those clauses of the sentence that contain the negation instead of removing the whole sentence. Then, the filtering of negations would take place after the treebanks are produced.

Second, *ActivityMiner* is capable of identifying the activities with high precision and accuracy. We have though to make the distinction between an activity and an activity's name. For example "carry_out removal_of_topsoil" is a valid activity while its name is not necessary in the most appropriate form; it should be "remove topsoil". The automatic renaming of the activities is a possibility to be explored. Another option would be to provide a way for the users to change the names of the activities manually. We still obtain false positives and false negatives mainly because of several reasons:

- The taggers and parsers do not work 100% correctly. For instance, when there are numbers in the sentence multiple errors appear;
- The verbs followed by a preposition pose problems. We are not able to tell at the moment if the construction verb + preposition is a transitive idiom (e.g. "look into the database") or not (e.g. "upload to the site"). Currently, these structures are considered as activities, exception made when the parser marks the word after the verb as particle (e.g. "carry out").
- Some activities belong to another process than the described one. They cannot be identified automatically. Similar to the activity names, we want to provide the possibility for the users to manually remove the false positives.

Further, a noun phrase (NP) can contain determiners ("this", "that" etc.) or pronouns ("it" etc.) instead of nouns. Currently, the determiners are replaced with the last encountered noun and the pronouns are kept as they are. We want to improve this feature in order to better identify the object referred by a determiner or by a pronoun.

Third, the authors, considering their thorough experience in process modeling, decided the reasoning rules of *ActivityRelationshipMiner*. An initial validation was made during one case. Other reports should be added to the corpus in order to check the existing rules and to discover new ones. The ordering of the sentences should also be better handled (we automatically consider them as sequential in this work).

Finally, lower priority was given to matters related to the performance or usability of the solution, but this is included in our future works.

V. VALIDATION AND PRELIMINARY EVALUATION

As specified in the Methodology section, we designed an evaluation to analyse the proposed technique in a practical case. We used the methodology section of the Villa Magna archaeological project [22, 23].

A. *Validation and Evaluation Setup*

The protocol is designed according to the main objective: discover process models from textual methodologies.

First, we wanted to validate the obtained logs - the list of activities (the partial output in Fig. 1). We targeted the correctness [35], completeness [36] and soundness [37] of the activities. The correctness allows us to know if all activities are well identified in the log. The completeness determines if the set of activities in the log capture all the activities from the report's methodology or from the real process. Finally, the soundness assesses the degree to which the log reflects the process activities described in the report.

Secondly, we wanted to validate the process instance models created from the log (the final output in Fig. 1). We used the final output to draw a process model manually, using the BPMN formalism [38]. Since we evaluated the obtained process model and not a modeling formalism, we consider we did not introduce any bias in this step.

Then, we focused on the preliminary evaluation of the technique: the correctness, comprehensiveness and utility of the process instance model, from the point of view of the real authors of the texts. They are specialists in archaeology and potential users of the presented approach. The comprehensiveness allows us to know if these potential users are able to understand the activities presented in the process instance model and their relationships. We also wanted to evaluate the whole approach in terms of its utility: does it allow the specialists to share knowledge with colleagues and make decisions based on the created methodological model? Nielsen [39] considers that utility is "synonymous with relevance or efficacy". In this particular case, we wanted to evaluate if the model allows the specialists to achieve a better understanding of the followed archaeological methodology.

According to other works in textual and discourse analysis, a sound evaluation should involve the participation of the authors of the texts themselves in order to avoid erroneous interpretations or bias [40]. Thus, the subject of this evaluation was the author of the report, being also one of the archaeologists responsible for the excavation works.

We designed a protocol consisting of six steps:

- Fill in the first part of the first questionnaire with personal background information;
- Read the report in order to recapitulate the archaeological project and the specific text;
- Access the log generated from the report;
- Fill in the second part of the first questionnaire that contains the log's validation;
- Access the process instance model drawn according to the generated model and analyse it;
- Fill in the second questionnaire, validating and evaluating the process instance model.

The questionnaires were available online to provide easy access to the author of the Villa Magna report. The first part of the first questionnaire requires personal information about the participant: the name and current affiliation, his/her professional background and the number of years of archaeological experience. The second part of the first questionnaire comprises questions regarding the correctness, completeness and soundness of the activity log: the activities that are correctly/incorrectly identified or named (true/false positives), the activities that are missing from the log but were described in the report's methodology; the extent to which the log reflects the process described in the report. The second questionnaire contains questions concerning the correctness, comprehensiveness and utility of the process instance model, its strengths, weaknesses and possible improvements. All questions are single-answer accompanied by a textbox that enables the author to freely express his/her opinions, and, for us, to extract qualitative information.

B. *Results*

Using *TextProcessMining* on the methodology text of the archaeology case, we obtained 34 activities. A fragment of the log is presented in Fig. 2. Out of these, the author reported in the first questionnaire:

- 4 false positives: "9.plan unit", "10.print transparent polyester sheet", "11.use millimetre grid" and "24.find deposit";
- 3 activities with wrong names: "11.model millimetre grid" instead of "model sheets", "13.allow recording of sequence" instead "allow recording of overlays" and "14.keep find" instead of "keep finds";
- 1 activity missing: "wet sieving".

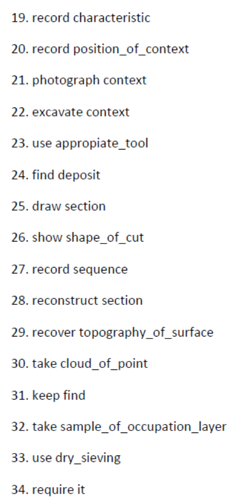
19. record characteristic
20. record position_of_context
21. photograph context
22. excavate context
23. use appropiate_tool
24. find deposit
25. draw section
26. show shape_of_cut
27. record sequence
28. reconstruct section
29. recover topography_of_surface
30. take cloud_of_point
31. keep find
32. take sample_of_occupation_layer
33. use dry_sieving
34. require it

Fig. 2. Fragment of the Discovered Log

The achieved precision is very satisfactory for the validation case with 88% correctly discovered activities. Regarding the missing activity, though the statement of the author is totally valid, the methodology text does not contain any sentences regarding it. The author assumes previous archaeological knowledge to reason about the convenience of choosing a wet or a dry sieving in function of specific criteria. Consequently, we are able to extract activities explicitly referred in the text, but we are not able to discover implicit activities.

Further, the naming of the activities could be improved first by improving the objects' identification and second by not lemmatizing the objects. In conclusion, the correctness reflected by the precision of the log is very satisfactory. The completeness is also very good, just one activity being reported as missing from the real process. In the final solution, the user will be able to review the log, exclude the false positives or add missing activities. Regarding the soundness, the author finds the real process activities "adequately" reflected in the log.

Concerning the relationships between activities, the archaeologist agrees with the discovered process instance model (a fragment is presented in Fig. 3). However, as she states: "what does not emerge is the situation of conditionality". The mutual exclusion is identified in multiple cases but not always (correctly). For example, "25.draw section" happens only in some situations in order to execute the activity "26.show shape of cut" and "27.record sequence" happens only under other situations when "25.draw section" is not required. An appropriate re-modelling could be summarised thus: if certain situations then "25.draw section" → "26.show shape of cut" else "27.record sequence". Similarly, in the process fragment below (Fig. 3), the activity "28.reconstruct section" does not happen all the times but only under certain circumstances. Another error concerns the activity "33.use dry sieving" which should come after the activity "22.excavate context". However, the archaeologist acknowledged it was a consequence of the order the activities were reported in text.

Regarding the overall comprehensiveness of the process instance model, the archaeologist is positive about its strengths and states that it "flows like the archaeological site investigation process". However, although the archaeologist reports that the model reflects adequately the report's methodology, she states she would not use the model for knowledge sharing with other colleagues or for process guidance. She would only use the model for teaching or for disseminating knowledge to non-specialists. These results about the utility of the process instance model might be grounded in the traditional way of working in Humanities but a more thorough evaluation is required from our side too.

VI. RELATED WORKS

Automatic analysis of text for discovering models has been researched for decades. Extensive work done in this field is related to requirements engineering where textual specifications are often created before the design and construction of any information system. Years ago, Rolland [41] identified four types of strategies to address the relationship between text and conceptual modelling: support the generation of models from text, support the model paraphrasing, help the general understanding of texts and improve text quality. Our work is related to the first strategy as we aim at discovering process models from textual methodologies. In a later work, Rolland [42] classifies the current techniques to support the discovery of conceptual models form text considering the following model-related characteristics: static or dynamic, rule-based or ontology-based. Regarding the static aspects of the conceptual models, there are tools to discover and generate object models or class models from textual requirements specifications [43, 44, 45, 46]. Regarding the dynamic aspects, Rolland highlighted works to extract use cases and scenarios from texts [47, 48, 49, 50]. Finally, regarding the rule-based or ontology-based aspects, there are approaches looking for discovering business process models from business rules [51, 52].

In addition to the later mentioned works, we have found other approaches to discover processes from: clinical documents [13], emails [14], stories extracted from collaboration [15] and business process documentation [16, 17]. Compared to our technique, [13] is domain dependent and fully-supervised, relying on prior annotation of the corpus. [14, 15] use a technique similar to ours but the key elements of an activity in their case are the actors and the verbs. The authors use a template for activity extraction that does not appear to handle passive-voice sentences and according to the exhibits in the articles there is no fine-grained identification of the objects. [16] first defines a list of templates that are used for identifying process-related sentences; second, they extract activities. No technical details are given and no indication of discovering a full process is found but rather process fragments. Finally, the solution proposed by [17, 18] is probably the closest to ours. The difference relies in the understanding of the verb semantics: we handle the passive/active sentences implicitly;

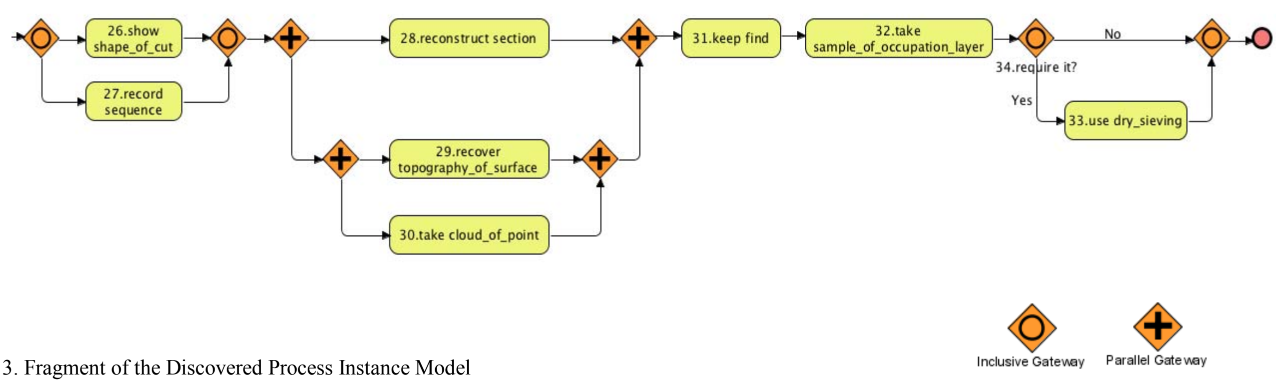

Fig 3. Fragment of the Discovered Process Instance Model

we avoid false positives by enforcing the discovery of only transitive verbs while the authors use a manually defined list of "weak verbs"; we can extract activities from treebanks directly while the authors rely on the grammatical relationships exposed as Stanford Dependencies [53]. Nonetheless, [17, 18] provide a mechanism for associating the pronouns to the corresponding concepts.

In other fields, complementary to Information Systems, there are works to identify statistical models from textual sources, with applications in biomedicine [54, 55]. The resulting models are essentially mathematical. Consequently, they lack the semantic richness achieved through conceptual models approaches, as previously explained.

We have identified also attempts to apply natural language processing techniques for text analysis in Humanities and Social Sciences disciplines, including archaeology [56]. However, these approaches mainly rely on previous annotation of the texts by experts or require the use of auxiliary domain-dependent ontologies.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed an automatic technique to extract activity logs and mine process instance models from textual methodologies. We have used natural language processing techniques focusing on the verb semantics for activities mining and a rule-based system for activity relationships mining. The validation and preliminary evaluation in an Archaeology case show that:

- The majority of the discovered activities are correct;
- The mined process instance model is satisfactory in comparison to the real enacted process;
- The archaeologist is positive concerning the comprehensiveness of the mined model but she is reserved regarding its usability for knowledge sharing or process guidance.

Although the results are promising, the technique can be improved to overcome its limitations. We highlight the following improvement areas: the identification of the negations; the activity discovery with idiomatic verbs; the activity naming; the ordering of the sentences and independent clauses taking in consideration the time clues; the rule-based system especially the mutual exclusion; the performance and usability. Additionally, we want to enrich the model by discovering other relationships such as iterations or implicit mutual exclusion and other information regarding an activity such as the actor.

We might also generate a XES log [57] instead of a log text file in order to enable the workers to load the file in ProM [58] or Disco [59]. In this way, the workers would be provided with an interactive application, which could support the process review, by deleting or renaming activities, and its representation using different formalisms. However, the main issue regarding the generation of the XES logs is the conservation of the activity relations captured from text. A possible solution to this matter would be the artificial injections of events and traces in the log.

As we mentioned in the beginning, this paper presents only the first step of our research aiming at mining a process instance model. However, in the second step we want to be able to mine multiple process instances from the same or different text and to aggregate those in one process model. This requires: (1) a method to identify if two process instances refer to the same process (2) a method to identify if two activities are equivalent or related (3) a method to aggregate two or more process instances of the same process. Regarding phase (3) there are already mature approaches dealing with process variability management and process model similarity, matching and merging in process model repositories [59, 60] that might be reused and adapted.

Finally, an extensive validation and evaluation including other domains apart from Archaeology (Medicine, Business, Information Systems etc.) and other data sources (business reports, ISO standards, software documentation etc.) is necessary in order to establish the generalization and usefulness of the proposed technique. The challenge will be to handle the threats to validity considering the characteristics of various domains and the background of the various workers following processes.

ACKNOWLEDGEMENT

We would like to thank the Villa Magna project's archaeologist for participating in the preliminary evaluation of *TextProcessMiner*.

REFERENCES

[1] A. Chrysanthi et al., *Thinking beyond the tool*. Oxford: Archaeopress, 2012.

[2] M. Gold, *Debates in the digital humanities*. Minneapolis: Univ Of Minnesota Press, 2012.

[3] P. Martín-Rodilla, "Empirical Approaches to the Analysis of Archaeological Discourse", in *Across Space and Time: 41st Computer Applications and Quantitative Methods in Archaeology Conference*, Perth, Australia, 2013.

[4] C. Gonzalez-Pérez and C. Parcero-Oubiña, "A Conceptual Model for Cultural Heritage Definition and Motivation" in *Revive the Past: Proceedings of the 39th Annual Conference on Computer Applications and Quantitative Methods in Archaeology*. Amsterdam University Press. Beijing, China, 2011, pp.234-244.

[5] C. Gonzalez-Perez, P. Martín-Rodilla, C. Parcero-Oubiña, P. Fábrega-Álvarez, and A. Güimil-Fariña, "Extending an Abstract Reference Model for Transdisciplinary Work in Cultural Heritage" in *Metadata and Semantics Research*, Springer Berlin Heidelberg, 2012, pp. 190-201.

[6] C. Hug, C. Salinesi, R. Deneckere, and S. Lamasse, "Process modeling for Humanities: tracing and analyzing scientific processes" in *Revive the Past: Proceedings of the 39th Annual Conference on Computer Applications and Quantitative Methods in Archaeology*. Amsterdam University Press. Beijing, China, 2011, pp.245-255.

[7] A. Tiwana, *The knowledge management toolkit: practical techniques for building a knowledge management system*. Upper Saddle River, NJ: Prentice Hall PTR, 2000.

[8] S. Brinkkemper, "Method engineering: engineering of information systems development methods and tools", *Information and Software Technology*, vol. 38, no. 4, pp.275-280, 1996.

[9] W. Aalst, *Process mining discovery, conformance and enhancement of business processes*. Berlin: Springer, 2011.

[10] Archaeopress.com, "Archaeopress - British Archaeological Reports", 2015. [Online]. Available: http://www.archaeopress.com/ArchaeopressShop/Public/defaultAll.as

p?intro=Home&PublishedDateGT=15+Feb+2014. [Accessed: 15- Feb- 2015].

[11] E. V. Epure, C. Hug, R. Deneckère, and S. Brinkkemper, "What Shall I Do Next? Intention Mining for Flexible Process Enactment" in *6th International Conference on Advanced Information Systems Engineering (CAiSE)*, 2014, Thessalonique, Greece. Springer International Publishing, 8484, pp.473-487.

[12] G. Khodabandelou, C. Hug, R. Deneckère, and C. Salinesi, "Unsupervised discovery of intentional process models from event logs" in *Proceedings of the 11th Working Conference on Mining Software Repositories*, May 2014, Hyderabad, India. ACM New York, NY, USA, pp.282-291.

[13] C. Thorne, E. Cardillo, C. Eccher, M. Montali, and D. Calvanese, "Process Fragment Recognition in Clinical Documents," in *13th Conference of the Italian Association for Artificial Intelligence (AI*IA 2013)*, Springer, 2013, pp.227-238.

[14] D. Soares, F. Santoro and F. Baião, "Discovering collaborative knowledge-intensive processes through e-mail mining", *Journal of Network and Computer Applications*, vol. 36, no. 6, pp.1451-1465, 2013.

[15] J. C. de A.R. Gonçalves, F. Santoro and F. Baião, "A case study on designing business processes based on collaborative and mining approaches," in 14th International Conference on *Computer Supported Cooperative Work in Design (CSCWD),* 2010, pp. 611-616.

[16] A. Ghose, J. Koliadis and A. Chueng, "Process Discovery from Model and Text Artefacts," in *IEEE Congress on Services,* 2007, pp.167-174.

[17] F. Friedrich, J. Mendling and F. Puhlmann, "Process Model Generation from Natural Language Text" in *23rd International Conference, CAiSE 2011, London, UK, June 20-24*, 2011, pp. 482-496.

[18] F. Friedrich, "Automated Generation of Business Process Models from Natural Language Input", M.Sc., School of Business and Economics. Humboldt-Universität zu Berli, 2010.

[19] R.J. Wieringa and A. Morali, "Technical Action Research as a Validation Method in Information Systems Design Science," in K. Peffers and M. Rothenberger and B. Kuechler (eds.) *Seventh International Conference on Design Science Research in Information Systems and Technology (DESRIST),* Springer, 2012, pp. 220-238.

[20] R. Yin, *Applications of Case study research*. Thousand Oaks, Sage Publications, 2003.

[21] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research", *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45-77, 2007.

[22] E. Fentress, "Archaeological Fieldwork Reports: Excavations at Villa Magna, 2009", Papers of the British Schoolat Rome, vol. 78, pp. 324-325, 2010.

[23] Villa-magna.org, "Villa Magna", 2015. [Online]. Available: http://www.villa-magna.org. [Accessed: 15- Feb- 2015].

[24] M.P. Marcus, M.A., Marcinkiewicz and B. Santorini, "Building a large annotated corpus of English: The Penn Treebank" *Computational linguistics*, vol. 19pp. 313-330, 1993.

[25] Ling.upenn.edu, "Penn Treebank P.O.S. Tags", 2015. [Online]. Available: https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html. [Accessed: 15- Feb- 2015].

[26] Python.org, "Welcome to Python.org", 2015. [Online]. Available: https://www.python.org. [Accessed: 15- Feb- 2015].

[27] Nltk.org, "Natural Language Toolkit — NLTK 3.0 documentation", 2015. [Online]. Available: http://www.nltk.org. [Accessed: 15- Feb- 2015].

[28] D. Klein and C. D. Manning, "Accurate Unlexicalized Parsing," in *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, 2003, pp. 423-430.

[29] Pypi.python.org, "pyenchant 1.6.6 : Python Package Index", 2015. [Online]. Available: https://pypi.python.org/pypi/pyenchant/. [Accessed: 15- Feb- 2015].

[30] K. Toutanova and C. D. Manning. "Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger," in *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000)*, 2010, pp. 63-70.

[31] K. Kipper, B. Snyder, and M. Palmer, "Extending a verb-lexicon using a semantically annotated corpus," in *Fourth International Conference on Language Resources and Evaluation (LREC 2004).* Lisbon, Portugal, May, 2004.

[32] G. Miller, "WordNet: a lexical database for English", *Communications of the ACM*, vol. 38, no. 11, pp. 39-41, 1995.

[33] A. X. Chang and C. D. Manning, "SUTIME: A Library for Recognizing and Normalizing Time Expressions," in *8th International Conference on Language Resources and Evaluation (LREC 2012), pp. 3735-3740.*

[34] P. Muller and A. Reymonet, "Using inference for evaluating models of temporal discourse," in *12th International Symposium on Temporal Representation and Reasoning, 2005. TIME 2005.* pp.11-19.

[35] B. Meyer, *Object-oriented software construction*. New York: Prentice-Hall, 1988.

[36] S. Rinderle, "Correctness criteria for dynamic changes in workflow systems - a survey" *Data & Knowledge Engineering* 50 (1), pp. 9-34, 2004.

[37] G. Boolos, J. Burgess and R. Jeffrey, *Computability and logic*. Cambridge: Cambridge University Press, 2002.

[38] P. Wong and J. Gibbons, "Formalisations and applications of BPMN", *Science of Computer Programming*, vol. 76, no. 8, pp. 633-650, 2011.

[39] J. Nielsen, *Usability engineering*. Boston: Academic Press, 1993.

[40] J. R. Hobbs, "On the Coherence and Structure of Discourse," Center for the Study of Language and Information (CSLI), Standford, CA, 1985.

[41] C. Rolland and C. Proix, "A natural language approach for Requirements Engineering" in *4th International Conference on Advanced information systems engineering CAiSE '92 Manchester*, UK, May 12–15, 1992, pp. 257-277.

[42] C. Rolland, "Conceptual Modeling and Natural Language Analysis" in *Seminal Contributions to Information Systems Engineering*, 2013, pp. 57-61

[43] L. Mich, "NL-LOOPS : From natural language to object oriented requirements using the natural language processing system LOLITA," *Natural Language Engineering*, Cambridge Universal, vol. 2, pp. 161-187, 1996.

[44] A. Moreno, "Object-Oriented Analysis from Textual Specifications," in *9th Conerence. on Software Engineering and Knowledge Engineering*, SEKE, 1997.

[45] N. Juristo, J. Morant and A. M. Moreno, "A formal approach for generating oo specifications from natural language", *Journal of Systems and Software*, vol. 48, no. 2, pp. 139-153, 1999.

[46] H.M. Harmain and R. Gaizauskas, "CM-Builder :An Automated NL-based Case Tool," in *15th International Conference on Automated Software Engineering. ASE'00*, 2000, pp. 45-54.

[47] C. Rolland and C. Ben Achour, "Guiding the construction of textual use case specifications," *Data & Knowledge Engineering*, vol.25, pp. 125-160, Elsevier, 1998.

[48] V. Menck, "Deriving behavior specifications from textual use cases," in *Proceedings of Workshop on Intelligent Technologies for Software Engineering (WITSE04, Sep 21, 2004, part of ASE 2004),* ed Linz, Austria, 2004, pp. 331-341..

[49] L. Kof, "Scenarios: Identifying Missing Objects and Actions by Means of Computational Linguistics," in *Proceedings of the 15th Int'l Requirements Engineering Conference (RE 2007)*, 2007. pp. 211-130.

[50] J. Santos et al., "Generating Requirements Analysis Models from Textual Requirements," in *First International Workshop on*

Managing Requirements Knowledge. MARK2008, IEEE 2008, pp. 32-41.

[51] H. Herbst, *Business rule-oriented conceptual modeling*. (Physica Verlag). Springer, Heidelberg, 1997.

[52] P. Rayson, L. Emmet, R. Garside, and P. Sawyer, “The REVERE Project: Experiments with the application of probabilistic NLP to Systems Engineering,” in *M. Bouzeghoub et al (eds.). Natural Language Processing and Information Systems*, pp. 288-300, Springer, Heidelberg, 2001.

[53] M. de Marneffe, B. MacCartney, and C. D. Manning, “Generating Typed Dependency Parses from Phrase Structure Parses” in Proceeding of *LREC*, 2006. pp. 449-454.

[54] W. Sun, A. Rumshisky and O. Uzuner, “Annotating temporal information in clinical narratives”, *Journal of Biomedical Informatics*, vol. 46, pp. S5-S12, 2013.

[55] G. Alterovitz and M. Ramoni, *Knowledge based bioinformatics*. Chichester, West Sussex: John Wiley & Sons, 2010.

[56] M. di Buono, M. Monteleone, P. Ronzino, V. Vassallo, and S. Hermon, "Decision making support systems for the Archaeological domain: A Natural Language Processing proposal," in *Digital Heritage International Conference (DigitalHeritage),* Marseille, France, 2013, pp. 397-400..

[57] Promtools.org, “start | ProM Tools”, 2015. [Online]. Available: http://www.promtools.org/doku.php. [Accessed: 15- Feb- 2015].

[58] Fluxicon.com, “Process Mining and Automated Process Discovery Software for Professionals - Fluxicon Disco.”, 2015. [Online]. Available: http://fluxicon.com/disco/. [Accessed: 15- Feb- 2015].

[59] M. La Rosa, et al., “Detecting approximate clones in business process model repositories,” *Information Systems*, vol. 49, pp. 102-125, 2015..

[60] M. La Rosa, et al., “APROMORE: An advanced process model repository,” *Expert Systems with Applications*, vol. 38, pp. 7029-7040, 2011.