Using Argumentation to Explain Ambiguity in Requirements Elicitation Interviews

Yehia Elrakaiby[‡], Alessio Ferrari[*], Paola Spoletini[‡], Stefania Gnesi[*] and Bashar Nuseibeh[‡§]
[*] CNR-ISTI, Pisa, Italy, Email: alessio.ferrari@isti.cnr.it, stefania.gnesi@isti.cnr.it
[†] Kennesaw State University, Marietta (GA), USA, Email: pspoleti@kennesaw.edu
[‡] LERO, University of Limerick (UL), Limerick, Ireland Email: yehia.elrakaiby@lero.ie, bashar.nuseibeh@lero.ie
[§] The Open University (OU), Milton Keynes, UK Email: b.nuseibeh@open.ac.uk

***Abstract*—The requirements elicitation process often starts with an interview between a customer and a requirements analyst. During these interviews, ambiguities in the dialogic discourse may reveal the presence of tacit knowledge that needs to be made explicit. It is therefore important to understand the *nature* of ambiguities in interviews and to provide analysts with cognitive tools to identify and alleviate ambiguities. Ambiguities perceived by analysts are sometimes triggered by specific categories of terms used by the customer such as pronouns, quantifiers, and vague or under-specified terms. However, many of the ambiguities that arise in practice cannot be rooted in single terms. Rather, entire fragments of speech and their relation to the mental state of the analyst need to be considered.**

In this paper, we show that particular types of ambiguities can be characterised by means of *argumentation theory*. Argumentation is the study of how conclusions can be reached through logical reasoning. In an argumentation theory, statements are represented as arguments, and conflict relations among statements are represented as attacks. Based on a set of ambiguous fragments extracted from interviews, we define a model of the mental state of the analyst during an interview and translate it into an argumentation theory. Then, we show that many of the ambiguities can be characterized in terms of 'attacks' on arguments. The main novelty of this work is in addressing the problem of explaining fragment-level ambiguities in requirements elicitation interviews through the formal modeling of the analyst's mental model using argumentation theory. Our contribution provides a data-grounded, theoretical basis to have a more complete understanding of the ambiguity phenomenon, and lays the foundations to design intelligent computer-based agents that are able to automatically identify ambiguities.

I. Introduction

Requirements elicitation is the process of gathering system requirements from stakeholders [1], [2], and can be performed through a variety of techniques, such as workshops, focus groups, scenarios and prototypes [3], [4]. Interviews with stakeholders are the most commonly used technique [5]–[8], and considered among the most effective ways to transfer knowledge [9]–[12]. Usually, requirements elicitation interviews involve a customer and a requirements analyst. The goal of the interview is to transfer the customer's knowledge and needs to the analyst, so that the latter can collect precise, correct and complete requirements, to be later conveyed to a requirements document. The success of an interview depends on several factors, such as the influence of domain knowledge [5], [13], [14]; the trustworthiness, motivation, and expressive ability of the customer [2]; the absorptive capacity and communication skills of the analyst [4], [5]; and ambiguities in the dialogic discourse [2]. Ambiguity in particular may play a negative or a positive role. Indeed, if the analyst cannot detect an ambiguity in the words of the customer, the knowledge transfer might be compromised [2]. Instead, a detected ambiguity can lead to the discovery of tacit knowledge [15], which is unexpressed, system-relevant information that needs to be elicited [16], [17]. For this reason, it is important to provide a framework to explain the ambiguity phenomenon and to give analysts the tools to recognise and alleviate ambiguity. In our previous work [18], we showed that some of the ambiguities occurring in interviews can be rooted in the nature of the *terms* used by the customer. In particular, we identified five categories of ambiguity cues, namely (1) under-specified, (2) vague, (3) quantifiers, (4) pronouns, and (5) domain-specific terms. However, about half of the ambiguity episodes analysed in our study could not be explained by focusing solely on the terms used by the customer. To understand these episodes, the mental context of the analyst, and its relation with the speech fragments of the customer need to be taken into account.

This paper aims to provide a theory for explaining ambiguity cases that cannot be rooted in single terms. To this end, we propose to use *argumentation theory* [19], [20] as a formal tool to show that these ambiguities can be represented as 'attacks' between arguments of a structured discourse that occurs in the mind of the analyst. By means of argumentation, we formalise one specific type of ambiguity, namely *acceptance unclarity* [21]. This phenomenon occurs when the analyst is not able to accept a speech fragment expressed by the customer, either because it is *inconsistent* with their current understanding of the problem space, or because it is *insufficient* to comprehend the problem.

We performed our study using an approach that is inspired by *grounded theory* [22]. Grounded theory entails an incremental process that focuses on a dialogue between the researcher and the data at hand, in which concepts are extracted from data, and theories are produced in terms of connections among concepts. In this work, we used data from 34 requirements elicitation interviews, involving domain experts and computer scientists. From this data, we isolated 232 customer's speech fragments that were classified as *ambiguous*. In the study presented in this paper, we focused

2332-6441/17 $31.00 © 2017 IEEE
DOI 10.1109/RE.2017.27

RE 2017, Lisbon, Portugal
Research Paper

on the subset of those fragments where the ambiguity is not triggered by a single term but by the entire fragment, and that caused the *acceptance unclarity* phenomena. We inspected those fragments and incrementally defined a theory based on argumentation to explain them. In this paper, we present our theory, together with examples taken from the interviews to show how the different categories of ambiguity can be explained by the theory. As with any qualitative study, the current contribution is biased by the background, vision and expectations of the authors. However, we believe that the reported examples provide sufficient evidence of the soundness of our approach. We believe that our contribution can be used as a theoretical basis to understand the nature of ambiguities in oral interviews. On the other hand, it can also act as a baseline for the development of intelligent computer-based agents that are able to detect particular types of ambiguities occurring during requirement elicitation interviews, as well as for detection of ambiguities in written requirements.

The remainder of the paper is structured as follows. In Sec. II we provide some background on ambiguity in interviews, and on argumentation theory. In Sec. III we describe the methodology followed in this study. Sec. IV presents our argumentation-based theory of ambiguity. Sec. V exemplifies the different categories of ambiguities and how they are represented and explained in the theory. Finally, Sec. VI discusses related works, and Sec. VII presents our conclusions.

II. Background

A. Ambiguity in Interviews

In our previous work [21], a categorisation of ambiguities in requirements elicitation interviews was provided. In this paper, we will focus on the ambiguity category named *acceptance unclarity*, which occurs whenever the analyst can understand the meaning of the customer's words, but cannot accept it for some reason. For the sake of space, we provide an informal definition only for this category. Formal definitions for all categories are available in Ferrari *et al.* [21].

To understand the phenomenon of acceptance unclarity, it is useful to provide a description of the process followed by the analyst to understand the customer's requirements. During an interview, the customer articulates *units of information*, i.e., system-related needs or domain-related knowledge. A unit of information is articulated by means of speech fragments, i.e., any consecutive set of words. Two main phases model the process of understanding of a speech fragment by an analyst: *interpretation* and *acceptance*. Interpretation is the phase in which the analyst gives a meaning to the speech fragment of the customer. Acceptance is the phase in which the analyst considers whether that meaning is *acceptable* with respect to their current understanding and knowledge of the problem space. With the term *acceptable* we mean that the speech fragment (a) appears sufficiently accurate to comprehend the problem, and (b) the analyst does not register any type of inconsistency with their current understanding of the problem, or with their knowledge of the domain. For example, consider the following speech fragment: *The onboard system of the train shall use a TCP/IP protocol to communicate*. The fragment can be interpreted, but it is not acceptable because it does not specify the device with which the system should communicate through the TCP/IP protocol. An analyst who hears the fragment would perceive an *insufficiency*, since the information that they have is insufficient to form a complete picture of the problem. Now, consider the speech fragment: *The train shall be able to stop within 5 meters, after an obstacle is detected on the line*. The fragment can be interpreted, but it is not acceptable by an analyst who knows that trains require hundreds of meters to stop when they are going at their full speed. In other terms, the analyst perceives an *inconsistency*.

An **acceptance unclarity** occurs whenever the analyst (a) can assign an interpretation to the speech fragment of the customer, (b) the interpretation matches the intended meaning of the customer, but (c) the interpretation is not acceptable. The cases of insufficiency and inconsistency exemplified above are both cases of acceptance unclarity.

B. Argumentation

Argumentation theory [23] is a form of reasoning that makes explicit the reasons for the conclusions that are drawn and how conflicts between reasons are resolved. This can provide a natural mechanism to handle inconsistent and uncertain information and to resolve conflicts of opinion between intelligent agents [20]. Argumentation in general has many uses [24] and is a particularly useful tool for the modeling of human dialog and phenomena such as negotiation and debate.

In argumentation theory, Dung's abstract argumentation framework [25] has been particularly influential, as it attempts to capture the essence of the process of reasoning about arguments and their acceptability, thus making it generally applicable across different application domains [26]. In its most simple form, the framework is a pair $\langle \mathcal{A}, \mathcal{D} \rangle$, where $\mathcal{A}$ is a set of arguments, which may be viewed as statements, and $\mathcal{D}$ is a set of attacks among those arguments. For example, an argument $A_1 \in \mathcal{A}$ stating that *speed will be measured using a laser device* can be attacked by another argument $A_2 \in \mathcal{A}$ stating that *no laser device may be used*. This attack is represented as $(A_1, A_2) \in \mathcal{D}$. Based on its arguments and attacks, argumentation frameworks enable the determination of which arguments are *acceptable* (or *justified*). This is performed through the calculation of the so-called *extensions* of argumentation frameworks.

Due to the abstract nature of Dung's framework, it is generally necessary to instantiate it before its application to a particular domain. As described in [26], the instantiation of Dung's framework typically consists of first transforming a description of the application domain (the argumentation inputs) (I), into an argumentation framework (AF), through the application of some function (f). Dung's machinery is then used to determine possible sets of acceptable arguments (AA) by calculating the *extensions* of AF, denoted by $\mathcal{EX}$(AF). The acceptable arguments are then mapped to outputs (O) through the application of another function (g). The four steps of this process are depicted in Fig. 1.

Fig. 1: Basic Argumentation System Overview

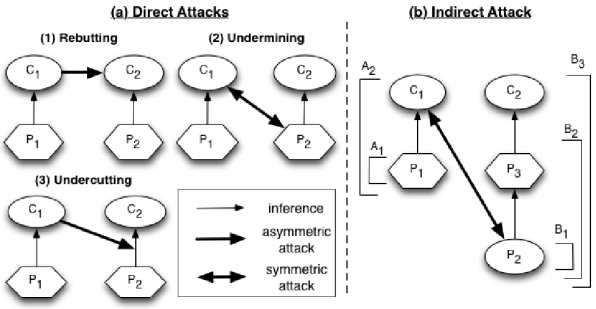

Fig. 2: Types of Attacks

The instantiation of arguments typically consists of providing arguments an internal structure in the form of inference rules. In this paper, we consider the $ASPIC^+$ framework [27] for structured argumentation. In $ASPIC^+$, basic arguments may correspond to simple propositions or to inference relations having the following elements:

- a set of assumptions or premises defining the conditions of the applicability of the argument,
- a conclusion representing the claim of the argument and,
- a method of deduction defining when the conclusion is entailed by the premises.

When arguments are described using the structure presented above, several attack types may be distinguished. Those attack types, depicted in Fig. 2, are briefly described as follows:

- Rebutting (Fig. 2-(a.1)): an argument (A_2) rebuts some argument (A_1) whenever the conclusion of A_1 cannot be true if the conclusion of A_2 holds.
- Undermining (Fig. 2-(a.2)): an argument (A_2) undermines some argument (A_1) whenever one of the premises of A_1 cannot be true if the conclusion of A_2 holds.
- Undercutting (Fig. 2-(a.3)): an argument (A_2) undercuts another argument (A_1) whenever the conclusion of A_1 attacks the inference relation of A_1.

An argument A *indirectly* attacks an argument B, if A attacks an argument that is (below B) in the inference tree of B. An example of an indirect attack is shown on the right hand side of Fig. 2 where A_2 indirectly attacks B_2 and B_3. Note that B_1 and B_2 are called the *proper* sub-arguments of B_3 and that A_2 is said to attack B_2 (or B_3) on (its sub-argument) B_1. Furthermore, an attack between two arguments A and B is called a *symmetric* attack if A attacks B and B attacks A. It is *asymmetric* if only A attacks B.

To summarize the working of $ASPIC^+$: starting from a knowledge base (or a set of propositions) and a set of inference rules, a set of arguments is generated in the form of inference trees, as depicted in Fig. 2. An attack relation is then derived from the structure of the generated arguments and a Dung's abstract argumentation framework is built. For example, the arguments in Fig. 2(b) generate the argumentation framework $\mathcal{AF} = \langle \mathcal{A}, \mathcal{D} \rangle$ where the set of arguments is $\mathcal{A} = \{A_1, A_2, B_1, B_2, B_3\}$ and the set of attacks is $\mathcal{D} = \{B_1 \leftrightarrow A_2\}$[1]. The extensions of $\mathcal{AF}$ are then calculated. Of interest to us in this paper are the *grounded* and *preferred* extensions. Intuitively, a grounded extension contains the arguments that are always justified, i.e. it represents the *skeptical* semantics of argumentation frameworks. A *preferred* extension is a (maximal) set of arguments that can be collectively accepted. Note that whenever an argument is not included in a preferred extension, then this argument must be in conflict with (or is attacked by) the extension. For example, $\mathcal{AF}$ has:

- the grounded extension $\{A_1\}$ and,
- the preferred extensions $\{A_1, A_2\}$ and $\{A_1, B_1, B_2, B_3\}$.

This can be interpreted as indicating that A_1 is always justified whereas either A_2 or $\{B_1, B_2, B_3\}$ may be accepted but not both. We say that an argument is *credulously* accepted if it belongs to at least one preferred extension. In this paper, a simplified version of $ASPIC^+$ is used where some features such as rebutting attacks and preferences between arguments are not considered and are left for future work.

III. Research Methodology

The methodology adopted for this study is inspired by grounded theory [22]. This is a qualitative research approach, in which the researcher analyses the data, and, through an iterative process of continuous confrontation with the data, develops a theory that explains it. Our approach does not strictly follow the guidelines of grounded theory as described by Corbin and Strauss [22], but rather its spirit, in that each category that composes a theory should be *grounded* on data. Our deviation from grounded theory is mainly due to the type of phenomenon considered, and the means adopted to analyse it. Grounded theory focuses on social phenomena, while here we are interested on a *cognitive* phenomenon, i.e., ambiguity. In grounded theory, the data analysis is performed through a process of *coding*, in which codes are conceptual categories that explain the data. In our case, we elaborate the data through a formal framework, i.e., argumentation, which explains the data not just through categories, but by *mapping* the data to a formal model. Below, we provide the details of the data considered, and the methodology followed in our study.

a) Data: We used a dataset of 232 ambiguous speech fragments, which we isolated in our previous study [21]. The speech fragments were identified from a set of 34 unstructured interviews that we arranged to study the phenomenon of ambiguity. In all interviews, the 2nd author of the current study acted as the analyst, while the customers were played by different domain experts and computer scientists. More information about the data is available in our previous publications [18], [21]. For each fragment, the following information had been

[1]The notation of $(A \leftrightarrow B)$ is used as a shortcut for $(A, B), (B, A)$.

provided: a natural language description of the ambiguity phenomenon that occurred; the category of ambiguity according to Ferrari *et al.* [21] (interpretation unclarity, acceptance unclarity, multiple understanding, undetected incorrect disambiguation, detected incorrect disambiguation); the category of the term that triggered the ambiguity phenomenon in the fragment (under-specified, vague, quantifier, pronoun, domain-specific, or *Fragment* – in case the ambiguity could not be rooted in any specific term). For the current study, we selected only those fragments that belonged to the *acceptance unclarity* category, and for which the trigger of the ambiguity was *Fragment*. A total of 77 fragments was selected.

b) Methodology: The methodology applied in our study follows a series of iterations, which are described below.

Preliminary Theory Definition. The 1st author was given a first subset of 8 fragments from 6 interviews. He inspected them and interacted with the 2nd author to better understand the perception that the latter had had of the ambiguity cases – we recall that the 2nd author acted as analyst during the interviews. From this phase, the 1st author developed an initial version of the theory presented in this paper. The theory was composed of (1) a *method* for translating the ambiguity phenomena into an argumentation framework and (2) a set of *categories* of attacks that occur between arguments and can explain the ambiguity phenomena. The structure of the theory remained the same also in its final version.

Categories Assessment. Then, the 3rd author, together with the 2nd author, inspected all the fragments. They annotated them with natural language memos that clarified the ambiguity phenomena that occurred, in light of the theory developed by the 1st author. In addition, they annotated the fragments with the categories of attacks defined by the first version of the theory. The goal of this phase was to assess the fitness of the categories on the data. Then, they went back to the 1st author with fragments that did not fit the theory, and with recommendations to improve the categories.

Method Assessment. Meanwhile, the 1st author was given 6 fragments belonging to 1 interview, in which he applied the method for translating the ambiguity phenomena into an argumentation framework. The goal of this analysis was to asses the soundness of the method on a sample of the data. Based on this analysis, he provided recommendations to improve the method.

Theory Revision. The theory was revised to consider the recommendations on the categories, and on the method. This has resulted in an improved theory that is able to cover and explain additional types of ambiguity. These iterations were useful as they lead to the successive identification of the different types of elements and relations of the model described in Sec. IV-A, and the constraints in Sec. IV-C.

Theory Assessment. The 2nd and 3rd author re-annotated the fragments according to the new categories of the theory, to re-assess their fitness. The annotation was reviewed by the 1st author. Then, a sample of 7 representative fragments was selected to be carefully analysed by the 1st author, who applied the final method on these fragments to validate its soundness. Those cases are the ones presented in this paper.

IV. An Argumentation-based Method

In oral interviews, analysts try to construct a mental model of the conversation based on information provided by customers and certain assumptions made by the analyst. The model includes both information about the system to be built and about the application domain since, during the interview, both aspects are discussed. The ambiguities on which we focus are episodes of *acceptance unclarity*, as defined in Sec. II. In this section, we first present the elements and relations of the proposed mental model of the analyst. Those elements and relations encode the analyst's current understanding of the problem space. Each time the analyst hears a speech fragment, they interpret it and confront this interpretation with their mental model of the system to be built. Conflicts that arise in this phase are acceptance unclarity cases. We then show how this model can be translated into an argumentation theory so that we can reason about the model and make a characterization of different acceptance unclarity cases.

A. Model Elements and Relations

The analyst's mental model $\mathcal{M}$ includes two sets: a set of elements $\mathcal{E}$ and a set of relations $\mathcal{R}$.

a) Elements Types and Representation: The set of elements $\mathcal{E} = \mathcal{E}_s \cup \mathcal{E}_m \cup \mathcal{E}_r$ is composed of the following types:

- **Statement** ($\mathcal{E}_s$): statements can be *optative*, typically when they refer to system requirements, or *indicative*, typically when they refer to domain aspects [28].
- **Motivation** ($\mathcal{E}_m$): description of a rationale element for a statement. The motivation answers *why* questions about a statement. It can explain the goal for a requirement, or provide the motivation for a domain aspect.
- **Realisation** ($\mathcal{E}_r$): description of a solution element to realise a statement in practice. The realisation answers *how* questions about a statement. It can be regarded as a specification for a requirement, or as a realisation of some domain aspect in the application domain.

Elements represent knowledge elements and are expressed in the form of simple atomic propositions, e.g., a motivation may be expressed as $product\_procurement$. Every element $e \in \mathcal{E}$ is associated to its type through the function $type : \mathcal{E} \rightarrow \{statement, motivation, realisation\}$, e.g., $type(product\_procurement) = motivation$ means that $product\_procurement$ is a *motivation* element. Whenever, a speech fragment is interpreted by the analyst, an associated knowledge element is added to their mind. Then, the analyst assigns a *type* to the newly collected element and constructs a model of the system based on this new knowledge element, considering the knowledge already gathered through previous interpretations of speech fragments, and also inferred or assumed knowledge elements. In this paper, we do not distinguish between explicit, implicit or inferred knowledge elements. Considering these aspects is possible, and would help assigning different degrees of confidence to knowledge

elements according to their type. However, this would unnecessarily complicate the model at this stage.

b) Relations Representation: The set $\mathcal{R}$ captures logical relations between different elements in the model. A relation is expressed in the form of an inference rule $r \in \mathcal{R}$ describing a logical connection between a set of elements $E \subseteq \mathcal{E}$, called the premises, and an element $e \in \mathcal{E}$, called the conclusion. Intuitively, a relation r means that the truth of the conclusion e depends on the truth of the premises E.

Relations are specified using expressions of the form $e_1, ..., e_n \Rightarrow e$ where $e, e_1, ..., e_n$ are elements. Relations state that when the premises, i.e., $e_1, ..., e_n$, are true, then, unless there is some evidence to the contrary, the conclusion, i.e., e, should hold.

B. Model Formalization in $ASPIC^+$

The $ASPIC^+$ framework [29] enables the definition of argumentation theories that include structured arguments and the generation of abstract argumentation frameworks [30], as discussed in Sec. II-B. A basic $ASPIC^+$ argumentation theory is composed of a knowledge base $\mathcal{KB}$, a set of inference rules $\mathcal{IR}$ and a logical language $\mathcal{L}$. Due to space limitations, we do not present the formal semantics of $ASPIC^+$ and refer the interested readers to Modgil and Prakken [27]. In this section, we briefly discuss the components of $ASPIC^+$ argumentation theories and describe how the analyst's mental model $\mathcal{M}$ can be represented as one.

a) Logical Language: Let $\mathcal{UE}$ be the universe of elements of $\mathcal{M}$, i.e., the set of all possible propositions from which elements of $\mathcal{E}$ can be drawn. The logical language composed of elements of $\mathcal{UE}$ and their negations is called the $ASPIC^+$ logical language of the model $\mathcal{M}$.

b) Knowledge Base: An $ASPIC^+$ knowledge base $\mathcal{KB} = \langle \mathcal{K}_n, \mathcal{K}_p \rangle$ consists of two disjoint subsets: $\mathcal{K}_n$ of *axiom* premises and $\mathcal{K}_p$ of *ordinary* premises. Ordinary premises represent *fallible* premises that can be attacked as opposed to premises in $\mathcal{K}_n$, which represent *axioms* that must always be true. Let $\mathcal{E}$ be the set of elements of $\mathcal{M}$, the knowledge base $\langle \emptyset, \mathcal{E} \rangle$ is called the $ASPIC^+$ knowledge base of $\mathcal{M}$. Notice that every element $e \in \mathcal{E}$ is included as an ordinary premise, i.e., as an assumption and not as a certain fact.

c) Inference Rules: The set of inference rules of an $ASPIC^+$ argumentation theory is a pair $\mathcal{IR} = \langle R_s, R_d \rangle$ where R_s is a set of *strict* inference rules and R_d is a set of *defeasible* inference rules. *Strict* inference rules are of the form $\phi_1, ..., \phi_n \rightarrow \phi$ and *defeasible* inference rules are of the form $\phi_1, ..., \phi_n \Rightarrow \phi$ where $\phi, \phi_1, ..., \phi_n$ are well-formed formulas of $\mathcal{L}$ [26]. Let $\mathcal{R}$ be the set of relations of $\mathcal{M}$. The set $\mathcal{IR} = \langle \emptyset, R \rangle$ is called the $ASPIC^+$ inference rules of $\mathcal{M}$. Notice that every $r \in \mathcal{R}$ is represented as a defeasible rule in $\mathcal{IR}$, i.e., their inferences may be attacked and withdrawn when there is an evidence to their contrary.

d) $ASPIC^+$ Argumentation theory: The argumentation theory of a model $\mathcal{M}$ is the theory that is composed of the $ASPIC^+$ logical language, knowledge base and inference rules of $\mathcal{M}$.

C. Model Constraints

This section defines four constraints that models of $\mathcal{M}$ should satisfy and describes how they can be incorporated in the argumentation theory of a model $\mathcal{M}$. These constraints arguably encode completeness and soundness requirements. They belong to two general categories: *satisfaction* constraints, and *realisation* constraints. The former encodes the idea that, when analysts hear a fragment that they interpret as a statement or a motivation, they shall also be able to mentally *refine* the content expressed in the fragment and identify a means to realise it. The latter encodes the idea that when analysts hear a fragment that they interpret as a statement or realisation, they shall be able to *abstract* from the fragment to understand the rationale behind it. Those refinement and abstraction constraints emulate the process that analysts follow in order to *accept* the different fragments that they hear.

a) Statement Satisfaction Constraint (SSC): this constraint means that every statement element, being it a domain aspect or a system requirement, has to be realisable, i.e., can be satisfied by some plausible set of realisation elements.

To check this constraint on the analyst's mental model $\mathcal{M} = \langle \mathcal{E}, \mathcal{R} \rangle$, we build an $ASPIC^+$ argumentation theory $\mathcal{AT}' = \langle \mathcal{KB}, \mathcal{IR}, \mathcal{L} \rangle$ as follows:

- $\mathcal{L}$ is the $ASPIC^+$ logical language of $\mathcal{M}$,
- $\mathcal{KB} = \langle \mathcal{K}_n, \mathcal{K}_p \rangle$ where $\mathcal{K}_n = \emptyset$ and $\mathcal{K}_p = \mathcal{E}_r$ where $\mathcal{E}_r$ is the set of realisation elements in $\mathcal{M}$, i.e., $\mathcal{E}_r = \{e \mid e \in \mathcal{E} \text{ and } type(e) = realisation\}$,
- $\mathcal{IR}$ is the set of of relations $\mathcal{R}$.

Let $\mathcal{E}_s = \{e \mid e \in \mathcal{E} \text{ and } type(e) = statement\}$ be the statement elements in $\mathcal{M}$. A statement $s \in \mathcal{E}_s$ is said to be *realisable* iff s can be credulously inferred from $\mathcal{AT}'$, i.e., s belongs to at least one of the preferred extensions of $\mathcal{AT}'$. The set $\{unrealisable(s) \mid s \in \mathcal{E}_s \text{ such as } s \text{ is not realisable}\}$ is called the set of statement satisfaction constraint elements generated from $\mathcal{M}$, denoted $\mathcal{E}_{ssc}$.

For example, let $\mathcal{E}_r = \{a, b, c\}$, $\mathcal{E}_s = \{s_1\}$ and $\mathcal{R} = \{\}$ be the realisation elements, statement elements and relations of $\mathcal{M}$ respectively. In this case, s_1 is not credulously inferred from the argumentation theory $\mathcal{AT}'$. Therefore, s is not realisable. In this case, the set of statement satisfaction constraint elements generated from $\mathcal{M}$ is $\{unrealisable(s_1)\}$.

Let $\mathcal{AT} = \langle \langle \mathcal{K}_n, \mathcal{K}_p \rangle, \langle R_s, R_d \rangle, \mathcal{L} \rangle$ be the argumentation theory of $\mathcal{M}$, the argumentation theory of $\mathcal{M}$ extended with statement satisfaction constraints is the theory $\mathcal{AT}_{ssc} = \langle \mathcal{KB}_{ssc}, \mathcal{R}_{ssc}, \mathcal{L}_{ssc} \rangle$ such that:

- $\mathcal{L}_{ssc}$ is the logical language $\mathcal{L}$ extended with the set $\{unrealisable(s) \mid s \in \mathcal{UE} \text{ and } type(s) = statement\}$ and their negations,
- $\mathcal{KB}_{ssc}$ is the knowledge base $\mathcal{KB}$ extended with the statement satisfaction constraint elements generated from $\mathcal{M}$ represented as axiom premises in $\mathcal{KB}_{ssc}$, i.e., $\mathcal{KB}_{ssc} = \langle \mathcal{K}_n \cup \mathcal{E}_{ssc}, \mathcal{K}_p \rangle$ and,
- $\mathcal{R}_{ssc} = \langle R_s \cup R_{ssc}, R_d \rangle$ where $R_{ssc} = \{unrealisable(s) \rightarrow -s \mid s \in \mathcal{E}_s\}$, i.e., the set of relations (inference rules) $\mathcal{R}$ is extended with a strict

rule $\{unrealisable(s) \rightarrow -s\}$ for every statement element of $\mathcal{M}$.

b) Motivation Satisfaction Constraint (MSC): this constraint means that every motivation element must be satisfiable in the model $\mathcal{M}$. More precisely, it encodes the notion that each motivation, being it a rationale for a domain aspect, or a system goal, shall be satisfiable by some set of statement elements. Violations of this constraint occur when the analyst finds that, given the current statements, a certain system goal is not satisfiable. Similarly, violations also occur when some domain-specific goal is not satisfiable by some set of domain aspects. This constraint can be defined in a similar way to statement satisfaction constraints and, therefore, will not be further discussed here for brevity.

c) Realisation Relevance Constraint (RRC): this constraint means that every realisation element must be relevant to the satisfaction of some statement. Violations of this constraint occur when the analyst cannot find a requirement that justifies the need for a practical solution element (or specification) that was suggested by the customer, or when the analyst cannot find a domain aspect that justifies the need for some practical realisation element. This constraint is defined as follows.

Let $\mathcal{E}_r = \{e \mid e \in \mathcal{E}$ *and* $type(e) = realisation\}$ be the set of realisation elements of $\mathcal{M}$ and $R' \subseteq \mathcal{E}_r$ be a subset of it. R' is called a *relevant realisation* set for a statement $s \in \mathcal{E}_s$ iff (1) s can be credulously inferred from the argumentation theory $\langle\langle \mathcal{K}_n, R'\rangle, \mathcal{R}, \mathcal{L}\rangle$ and s cannot be credulously inferred from any argumentation theory $\langle\langle \mathcal{K}_n, R''\rangle, \mathcal{R}, \mathcal{L}\rangle$ where R'' is a proper subset of R', i.e., when $R'' \subset R'$. Let $RR(s)$ be the union of all the relevant realisation sets for a statement $s \in \mathcal{E}_s$. A realisation element $r \in \mathcal{E}_r$ is said to be *irrelevant* iff it does not belong to any relevant realisation set of any statement, i.e., $\nexists s \in \mathcal{E}_s$ such that $r \in RR(s)$. The set $\{irrelevant(r) \mid r \in \mathcal{E}_r$ *and r is irrelevant*$\}$ is called the set of realisation relevance constraint elements of $\mathcal{M}$.

For example, let $\mathcal{E}_r = \{a, b, c, d\}$, $\mathcal{E}_s = \{s\}$ and $\mathcal{R} = \{(a, b, c \Rightarrow s), (a, b \Rightarrow s), (d \Rightarrow s)\}$, then the relevant realisation sets of s are $S_1 = \{d\}$ and $S_2 = \{a, b\}$ and $RR(s) = \{a, b, d\}$. If s is the only statement, then the realisation element c is irrelevant.

The argumentation theory of $\mathcal{M}$ extended with realisation relevant constraints, denoted $\mathcal{AT}_{rrc}$, can be defined in a similar way to the argumentation theory extended with statement satisfaction constraints.

d) Statement Relevance Constraint (SRC): this constraint means that every statement element must be relevant to the satisfaction of some motivation element. It encodes the notion that each requirement and domain aspect should contribute to the satisfaction of some motivation element. Violations of this constraint occur when the analyst cannot find a reason for the existence of some statement. The definition of this constraint is similar to the one of the realisation relevance constraint.

V. Categories of Ambiguity

Acceptance unclarity stems from two main sources, namely inconsistency and insufficiency. For the detection of inconsistencies, a model of the analyst's mental state $\mathcal{M}$ is defined, as described in Sec. IV-A. This model is then translated into its $ASPIC^+$ argumentation theory, as presented in Sec. IV-B. After arguments are generated from this argumentation theory, inconsistencies take the form of *symmetric* attacks between the generated arguments. On the other hand, insufficiencies take the form of *asymmetric* attacks when the argumentation theory of $\mathcal{M}$ is extended with constraints, as described in Sec. IV-C.

A. Inconsistency Categories

An inconsistency occurs whenever the analyst perceives a contradiction after the introduction of a knowledge element by the customer. Inconsistencies are detected when there are symmetric attacks between the arguments built on the basis of the argumentation theory of the analyst's model $\mathcal{M}$. In this section, the notation S (X, Y), where X,Y$\in \{m, s, r\}$, is used to represent an attack between an element of types X and another of type Y. For example, (S (m, s)) is an attack between a motivation and a statement element. Notice that this is an informal classification where an attack is said to be of type (X, Y) if the inconsistency is perceived after the introduction of an element of type X, leading to a situation where a choice has to be made between this element and another of type Y. According to this classification, there are 6 basic categories of inconsistency – given by the number of possible unordered pairs between element types. A more formal classification of attacks is left for future work.

Example V.1 (S (s, s))**.** One of our customers wants to develop a system to allow patients to measure the amount of glucose in their blood, and then send the result to their general practitioner. If the glucose level is above a certain threshold, the practitioner pays a visit to the patient. The customer says: *On the doctor's side, it (the system) is a PC program [...] In one-two days the doctor sees the notification* (s_1). The domain knowledge of the analyst tells him that the doctor might be on holiday (r_1). Hence, the notification might be severely delayed (inference relation i_1 with conclusion $s_2 = -s_1$). The analyst asks for clarifications. The customer replies: *The general practitioner is substituted by another doctor who accesses the same system*.

We model this example as a model $\mathcal{M}$ which includes:

- the elements $\mathcal{E} = \{s_1, r_1, s_2\}$,
- the relations $\mathcal{R} = \{r_1 \Rightarrow s_2\}$.

The arguments constructed on the basis of the argumentation theory of $\mathcal{M}$, depicted in Fig. 3(a), are:

- B_1: in one-two days the doctor sees the notification,
- A_1: the doctor might be on holiday,
- A_2: since the the doctor might be on holiday, the notification might be severely delayed.

The Dung's abstract argumentation framework AF generated from this argumentation theory consists of:

- the arguments: $\{A_1, A_2, B_1\}$ and,
- attacks: $\{A_2 \leftrightarrow B_1\}$ which intuitively means that there is a (symmetric direct) attack between B_1 and A_2.

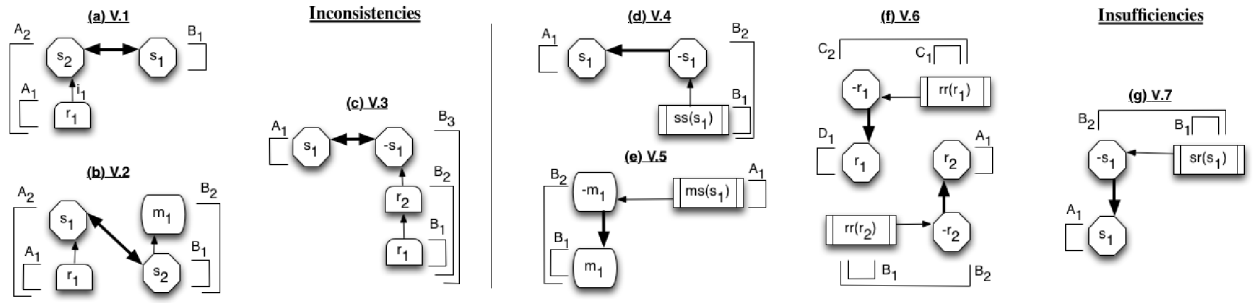

Fig. 3: Generated Arguments

The grounded and preferred extensions of AF are:

- the grounded extension: $\{A_1\}$ and,
- the two preferred extensions: $\{A_1, B_1\}$ and $\{A_1, A_2\}$.

Thus, the calculation of the extensions of this mental model reveals that A_1 can be accepted unconditionally, but one has to choose to accept either B_1 or A_2, i.e., either accept the relation $r_1 \Rightarrow s_2$, i.e., "*since the doctor might be on holiday, then the notification might be severely delayed*" or the statement s_1, i.e., "*in one-two days the doctor sees the notification*".

Example V.2 (S (m, s))**.** One of the customers wants a recycling-support system that, given the envelope of a product, indicates to the user which trash bin should be used. One of the system goals, made explicit during the interview, is *to avoid fines from the municipality for incorrect recycling* (m_1). According to the domain knowledge of the analyst, *trash bins are placed along the streets* (r_1), and *therefore there is no way to trace the owner of the rubbish, once it is thrown in the trash bin* (inference relation i_1 with conclusion s_1). This goal was in contrast with the domain knowledge of the analyst, who could not see how *the municipality identifies the person who violates the aforementioned rule* ($s_2 = -s_1$).

We represent this example as a model $\mathcal{M}$ which includes:

- the elements $\mathcal{E} = \{m_1, r_1, s_1, s_2\}$,
- the relations $\mathcal{R} = \{r_1 \Rightarrow s_1\}$.

The arguments generated from $ASPIC^+$ theory of $\mathcal{M}$, depicted in Fig. 3(b), are:

- A_1: trash bins are placed along the streets,
- A_2: since trash bins are placed along the streets, garbage cannot be traced back to their owner,
- B_1: people who do not recycle should be fined,
- B_2: to fine people, the municipality must be able to trace products in trash bins back to their owners.

The Dung's abstract argumentation framework AF corresponding to this argumentation theory consists of:

- the arguments: $\{A_1, A_2, B_1, B_2\}$ and,
- attacks: $\{A_2 \leftrightarrow B_1\}$.

The grounded and preferred extensions of AF are:

- the grounded extension: $\{A_1\}$ and,
- the two preferred extensions: $\{A_1, A_2\}$ and $\{A_1, B_1, B_2\}$.

Thus, the calculation of the extensions of this mental model reveals that A_1 is unconditionally acceptable, but, if one accepts A_2, then $\{B_2, B_1\}$ must be rejected, or the opposite.

Example V.3 (S (s, r))**.** One of the customers wants to develop an app to manage medical-related reservation (e.g., x-rays, specialists visits, blood tests, *etc.*) in Tuscany. The analyst assumed that the *current reservation system, based on phone calls, was centralised* (s_1). During the interview, this statement was attacked by the description of a realization given by the customer. More precisely, the customer says: *depending on where the examination/visit will be* (r_1), *[The patient has to call] Nottola or Siena* (r_2). The analyst could not understand how this realization was possible since *calling Nottola or Siena* (r_2) *means that the reservation system is not centralized* (inference i_2 with conclusion $-s_1$).

The arguments generated on the basis of the argumentation theory of the mental model of the analyst for this example are depicted in Fig. 3(c).

B. Insufficiency Categories

An insufficiency occurs whenever the analyst perceives that they need more information in order to accept a new knowledge element from the customer. Insufficiencies cannot be detected by directly inspecting the model as it requires a form of (meta) reasoning on the model. To enable this kind of reasoning, we labeled knowledge elements and introduced the different constraints in Sec. IV-C. Note that this approach is somewhat similar in spirit to the *meta-argumentation* approach proposed in [26]. However, the target argumentation system in this paper is $ASPIC^+$ and not Dung's argumentation framework. There are 12 categories of insufficiencies, given by the rank of the cartesian product between the number of constraints (4, namely SSC, MSC, RRC, SRC – acronyms are defined in Sect. IV-C) and the number of element types (3, namely m, s, r). Insufficiencies are revealed by asymmetric attacks in the argumentation theory of $\mathcal{M}$ when it is extended with constraints, as described in Sec. IV-C. In the following examples, the notation A (Z, X) where $Z \in \{SSC, RRC, MSC, SRC\}$ and $X \in \{m, s, r\}$ is used to represent asymmetric attacks between some knowledge element of type X and a constraint argument of type Z.

Example V.4 (A (SSC, s))**.** One of our customers is a mechanical engineer who wishes to develop a system to facilitate getting the quotes of mechanical components from different vendors. When speaking about the implementation of the system to get the quotes, he says: *I should have the name of the vendor, and an email address to contact* (optative statement s_1). The analyst could not understand how, in practice, this information could be retrieved. In other words, the analyst was not able to find a realisation for this statement.

This simple example can be represented as a model $\mathcal{M}$ which only includes the elements $\mathcal{E} = \{s_1\}$ and an empty set of relations $\mathcal{R} = \{\}$. The $ASPIC^+$ theory of $\mathcal{M}$ is consistent in the sense that there are no attacks between its arguments. In order to reveal this kind of ambiguities, we construct the $ASPIC^+$ argumentation theory with statement satisfaction constraints $\mathcal{AT}_{ssc}$ as described in Section IV-C. The arguments built on top of this argumentation theory are depicted in Fig. 3(d). which shows an asymmetric attack between the constraint B_2 and the unrealised statement A_1. This argumentation theory has only one and the same grounded and preferred extension, namely the extension $\{B_1, B_2\}$ which should be interpreted as follows:

- B_1: statement s_1 is not realisable,
- B_2: therefore s_1 cannot be accepted.

Example V.5 (A (MSC, s))**.** One of our customers is a physician who wishes to develop a system for automatically monitoring the diet of a representative sample of the population for research purposes. Currently, the diet of this sample is evaluated by means of a questionnaire, where people are asked how often do they eat meat, vegetables, fish, *etc.* The sample is randomly selected, and the customer says that the problem is that *people tell lies [about their diet]* (domain aspect s_1). Hence, one of the goals of the system should be to know exactly what people eat (motivation m_1). She suggests having a system in which people take pictures of their meals (requirement s_2). However, after some discussion, it became clear that this system could not address the previously agreed goal. In other terms, the analyst could not find a set of statements that allows satisfying the motivation of the system (therefore the satisfaction of the motivation is not entailed from the collected knowledge).

The application of the motivation satisfaction constraints to Example *V.5* produced the arguments depicted in Fig. 3(e). Note that if statement satisfaction constraints are verified, then s_1 and s_2 would be detected as *unrealisable*.

Example V.6 (A (RRC, r))**.** One of our customers wants to develop a Web-based platform to ease communication between citizens and representatives of the parliament. When speaking about the realisation of the idea, he makes the following scenario: *If I have many crimes in one area (r_1), and I start having many posts about security coming from that region (r_2), I can associate this [the crimes with the region] (s_1).* The analyst could not understand what was the requirement addressed by this scenario (the relation i_1 with premises $\{r_1, r_2\}$ and the conclusion s_1 is not accepted by the analyst), since the main requirement understood up to that moment was to allow citizens to send information to the representative of the parliament. In other terms, a statement that can be satisfied with the suggested realisation was missing.

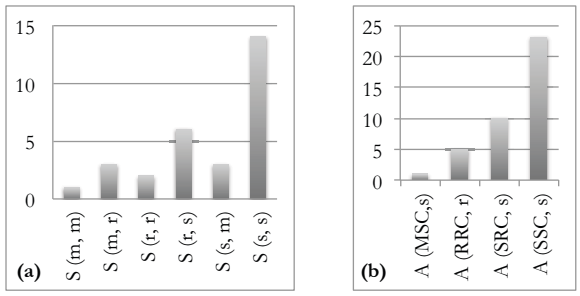

Fig. 4: Distribution of (a) symmetric and (b) asymmetric attacks.

The application of realisation relevance constraints to Example *V.6* produces the arguments depicted in Fig. 3(f).

Example V.7 (A (SRC, s))**.** One of our customers wants to develop a mobile application that allows him to know the closest and cheapest petrol station while he is driving (m_1). When speaking about the parameters that the application should take into account, the customer says: *Whether or not you're on the highway, which I think is also important* (s_1). The analyst could not understand the motivation of this statement (a relation between s_1 and previously stated motivations was not identified), since, in his experience, *fuel prices do not depend on whether you are on the highway or not* (r_1). After asking for clarifications, the customer specified that *It [the fuel] is cheaper for you if you exit the highway*.

The application of statement relevance constraints to *V.7* produces the arguments depicted in Fig. 3(g).

C. Statistics on the Data

Among the 77 analyzed fragments, 39 were categorised as symmetric attacks (inconsistency), 29 as asymmetric attacks (insufficiency) while 1 included both conflicting and missing aspects. The remaining 8 (around 12%) could not be represented within the framework: 4 cases were incorrectly classified as acceptance unclarities due to fragments, while they were rooted in terms; and the remaining 4 were insufficiency cases which detection requires the incorporation of additional constraints in the framework. This extension is left for future work. In Figure 4, we report the distribution of (a) symmetric and (b) asymmetric attacks over the different categories. In both cases, there is a category of attack that is much more frequent than the others. Specifically, symmetric attacks between statements (S (s, s)) represent almost half of the symmetric cases. Attacks between a statement and the constraint expressing the need to have a realisation for a statement (A (SSC, s)) represent more than half of the asymmetric cases.

VI. Related Work

This section discusses related work on ambiguity, inconsistency, insufficiency and argumentation.

a) Ambiguity: Ambiguity in natural language has been studied extensively in RE, especially in relation to its occurrence in written requirements. In particular, strategies were defined to *prevent* ambiguities by means of formal approaches [31]–[33] or constrained natural languages [34], [35]. Other approaches aim to *detect* ambiguities in requirements. These approaches are mainly *rule-based*, i.e., based on linguistic patterns to be matched within requirements [36]. Automated tools such as QuARS [37], SREE [38] and others [39], [40] were developed according to this philosophy. Other work [41], [42] focuses on the usage of *statistical* approaches to detect particular types of ambiguity cases, the so-called *innocuous ambiguities* – i.e., linguistic ambiguities that have one single reading in practice. All the cited work focuses on ambiguities in written texts that can be rooted in terms. Our work differs since we focus on ambiguities that depend on the context, referred to as *pragmatic* ambiguities [36].

b) Inconsistency: Inconsistency occurs when a requirements document contains conflicting, contradictory descriptions of the expected behavior of the system to be built or of its domain [43]. The majority of techniques developed to detect inconsistencies focus on the usage of formal logics [43] or models [44]–[46] to evaluate the overall consistency of formalised requirements, specifications and domain assertions. Tools for the detection of inconsistencies, such as EA-Analyser [47], were also developed. The majority of the cited work focuses on the analysis or negotiation phases of the RE process, when (part of) the requirements are already documented. Our work focuses on the *early* elicitation phase.

c) Insufficiency: In RE, insufficiency is explicitly studied by Pitts and Browne [48] in the context of requirements elicitation interviews. In particular, they studied how analysts with different degrees of expertise have different cognitive strategies to assess the sufficiency of the information that they receive from the customer. To our knowledge, insufficiency is not treated by other studies. However, insufficiency is closely related to the concept of requirements *completeness*. In a sense, insufficiency is the perception of some form of incompleteness in the requirements from the point of view of the analyst. Studies about requirements completeness provide several definitions of the concept [49]–[51]. In these studies, completeness is regarded as an objective property of the specification, although its evaluation requires domain expertise, e.g., to build a domain model against which the completeness of a specification has to be evaluated. Our paper emphasizes a *subjective* aspect of incompleteness. For this reason, in line with Pitts and Browne, we use the term insufficiency. However, while we focus on insufficiency of single fragments of information received, Pitts and Browne focus on the determination of the sufficiency of the overall information received during the interview.

d) Argumentation: Two forms of argumentation have been often considered for modeling and reasoning about RE artifacts: *Toulmin* arguments and *Dung*'s argumentation frameworks. One of the first works on argumentation in RE is that of Haley *et al.* [52] where *security satisfaction arguments* are proposed as a means to convince a reader that a system satisfies its security requirements. Franqueira *et al.* [53] extended this work and introduced a risk assessment method (RISA), which identifies rebuttals and mitigations needed to satisfy security requirements. Mirbel and Villeta [54] proposed an approach for the management of requirements artifacts based on argumentation-theory. Given goal-oriented requirements models, where goals are associated using different relations, an extended argumentation framework is generated and possible alternative sets of consistent requirements are identified. Ingolfo *et al.* [55] propose a five-step iterative process to systematically establish compliance of system requirements with law through discussions among stakeholders. Jureta *et al.* [56] proposed the ACceptability Evaluation Framework (ACE) to represent, in the form of a graph, information exchanged in a discussion about the relative validity of an RE artifact. Bagheri and Ensan [57] model interaction and inconsistencies between requirement statements using Dung's abstract argumentation framework. They also propose techniques to rank and select between the framework's *preferred* extensions, when more than one exists. In comparison, firstly, our aim is to model the phenomenon of ambiguities arising in oral interviews. Secondly, we consider $ASPIC^+$ for structured argumentation which builds on the foundational work of Dung. This arguably provides our modeling language a high-level of expressiveness and, at the same time, enables us to profit from a large amount of theoretical work in this highly active research field.

VII. Conclusion

Ambiguity in natural language is a complex phenomenon that has been studied by philosophers, linguists and computer scientists. However, most previous work on the topic has focused on ambiguities in written text, specifically, those caused by ambiguous natural language terms. Our work advances the state-of-the-art by focusing on ambiguities that occur in oral communication that cannot be rooted in single terms. We showed that argumentation theory can be used to explain these cases of ambiguity, which are particularly common in requirements elicitation interviews. Our future work will provide the basis to automate the method described in this paper. We are aware that a full automation would require advanced natural language processing (NLP) technologies that can perform semantic tasks, which are not currently available [38]. Hence, we will focus on identifying (a) the tasks of the method that can be automated, and (b) those that require human intervention, to come to a semi-automated process. Further automation of this process will be possible when the required NLP technologies are available.

Acknowledgment

This work was supported, in part, by Science Foundation Ireland grant 13/RC/2094 and ERC Advanced Grant 291652.

REFERENCES

[1] I. Sommerville and P. Sawyer, "Viewpoints: principles, problems and a practical approach to requirements engineering," *Annals of Software Engineering*, vol. 3, no. 1, pp. 101–130, 1997.

[2] A. Distanont, H. Haapasalo, M. Vaananen, and J. Lehto, "The engagement between knowledge transfer and requirements engineering," *IJKL*, vol. 1, no. 2, pp. 131–156, 2012.

[3] S. Robertson and J. Robertson, *Mastering the requirements process: Getting requirements right*. Addison-wesley, 2012.

[4] D. Zowghi and C. Coulin, "Requirements elicitation: A survey of techniques, approaches, and tools," in *Engineering and managing software requirements*. Springer, 2005, pp. 19–46.

[5] I. Hadar, P. Soffer, and K. Kenzi, "The role of domain knowledge in requirements elicitation via interviews: an exploratory study," *Requirements Engineering*, vol. 19, no. 2, pp. 143–159, 2014.

[6] R. Agarwal and M. R. Tanniru, "Knowledge acquisition using structured interviewing: an empirical investigation," *JMIS*, vol. 7, no. 1, 1990.

[7] G. J. Browne and M. B. Rogich, "An empirical investigation of user requirements elicitation: Comparing the effectiveness of prompting techniques," *JMIS*, vol. 17, no. 4, pp. 223–249, 2001.

[8] W. R. Friedrich and J. A. Van Der Poll, "Towards a methodology to elicit tacit domain knowledge from users," *IJIKM*, vol. 2, no. 1, 2007.

[9] A. Sutcliffe and P. Sawyer, "Requirements elicitation: towards the unknown unknowns," in *RE'13*. IEEE, 2013, pp. 92–104.

[10] A. Davis, O. Dieste, A. Hickey, N. Juristo, and A. M. Moreno, "Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review," in *RE'06*. IEEE, 2006, pp. 179–188.

[11] J. Coughlan and R. D. Macredie, "Effective communication in requirements elicitation: a comparison of methodologies," *Requir. Eng.*, vol. 7, no. 2, pp. 47–60, 2002.

[12] A. M. Hickey and A. M. Davis, "A unified model of requirements elicitation," *J. Manage. Inf. Syst.*, vol. 20, no. 4, pp. 65–84, Mar. 2004.

[13] A. M. Aranda, O. Dieste, and N. Juristo, "Effect of domain knowledge on elicitation effectiveness: An internally replicated controlled experiment," *TSE*, vol. 42, no. 5, pp. 427–451, 2016.

[14] A. Niknafs and D. M. Berry, "An industrial case study of the impact of domain ignorance on the effectiveness of requirements idea generation during requirements elicitation," in *RE'13*. IEEE, 2013, pp. 279–283.

[15] A. Ferrari, P. Spoletini, and S. Gnesi, "Ambiguity as a resource to disclose tacit knowledge," in *RE'15*. IEEE, 2015, pp. 26–35.

[16] M. Polanyi, *The Tacit Dimension*. Garden City, NY: Doubleday, 1966.

[17] V. Gervasi, R. Gacitua, M. Rouncefield, P. Sawyer, L. Kof, L. Ma, P. Piwek, A. De Roeck, A. Willis, H. Yang *et al.*, "Unpacking tacit knowledge for requirements engineering," in *Managing requirements knowledge*. Springer, 2013, pp. 23–47.

[18] A. Ferrari, P. Spoletini, and S. Gnesi, "Ambiguity cues in requirements elicitation interviews," in *RE'16*. IEEE, 2016, pp. 56–65.

[19] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artificial intelligence*, vol. 77, no. 2, pp. 321–357, 1995.

[20] S. Modgil and H. Prakken, "The ASPIC+ framework for structured argumentation: a tutorial," *Argument Comput.*, vol. 5, pp. 31–62, 2014.

[21] A. Ferrari, P. Spoletini, and S. Gnesi, "Ambiguity and tacit knowledge in requirements elicitation interviews," *REJ*, vol. 21, no. 3, pp. 333–355, 2016.

[22] J. M. Corbin and A. Strauss, *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory, 3e*. Sage, 2007.

[23] I. Rahwan and G. R. Simari, "Argumentation in artificial intelligence," *Argumentation Artif. Intell.*, vol. 171, pp. 1–493, 2009.

[24] S. E. Toulmin, "The Uses of Argumentation," *Cambridge Univ. Press*, vol. 37, no. 2, pp. 168–182, aug 1959.

[25] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *IJCAI*, vol. 77, no. 2, pp. 852–857, 1993.

[26] G. Boella, D. M. Gabbay, L. van der Torre, and S. Villata, "Meta-argumentation modelling I: Methodology and techniques," *Stud. Log.*, vol. 93, no. 2, pp. 297–355, 2009.

[27] S. Modgil and H. Prakken, "The aspic + framework for structured argumentation: A tutorial," 1 2014, vol. 5, pp. 31–62.

[28] M. Jackson, "The meaning of requirements," *Annals of Software Engineering*, vol. 3, no. 1, pp. 5–21, 1997.

[29] S. Modgil and H. Prakken, "A general account of argumentation with preferences," *Artif. Intell.*, vol. 195, pp. 361–397, 2013.

[30] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artif. Intell.*, vol. 77, no. 2, pp. 321–357, 1995.

[31] L. Kof, "From requirements documents to system models: A tool for interactive semi-automatic translation," in *RE'10*, 2010.

[32] V. Ambriola and V. Gervasi, "On the systematic analysis of natural language requirements with Circe," *ASE*, vol. 13, 2006.

[33] L. Mich, "NL-OOPS: from natural language to object oriented requirements using the natural language processing system LOLITA," *NLE*, vol. 2, no. 2, pp. 161–187, 1996.

[34] A. Mavin, P. Wilkinson, A. Harwood, and M. Novak, "Easy approach to requirements syntax (ears)," in *RE'09*. IEEE, 2009, pp. 317–322.

[35] C. Arora, M. Sabetzadeh, L. Briand, and F. Zimmer, "Automated checking of conformance to requirements templates using natural language processing," *TSE*, vol. 41, no. 10, pp. 944–968, 2015.

[36] D. M. Berry, E. Kamsties, and M. M. Krieger, "From contract drafting to software specification: Linguistic sources of ambiguity," 2003.

[37] S. Gnesi, G. Lami, and G. Trentanni, "An automatic tool for the analysis of natural language requirements," *IJCSSE*, vol. 20, no. 1, 2005.

[38] S. Tjong and D. Berry, "The design of SREE a prototype potential ambiguity finder for requirements specifications and lessons learned," in *REFSQ'13*, ser. LNCS, 2013, vol. 7830, pp. 80–95.

[39] B. Gleich, O. Creighton, and L. Kof, "Ambiguity detection: Towards a tool explaining ambiguity sources," in *REFSQ'10*, ser. LNCS, vol. 6182. Springer, 2010, pp. 218–232.

[40] H. Femmer, D. M. Fernández, S. Wagner, and S. Eder, "Rapid quality assurance with requirements smells," *JSS*, vol. 123, pp. 190–213, 2017.

[41] F. Chantree, B. Nuseibeh, A. N. D. Roeck, and A. Willis, "Identifying nocuous ambiguities in natural language requirements," in *RE'06*, 2006, pp. 56–65.

[42] H. Yang, A. N. D. Roeck, V. Gervasi, A. Willis, and B. Nuseibeh, "Analysing anaphoric ambiguity in natural language requirements," *Requir. Eng.*, vol. 16, no. 3, pp. 163–189, 2011.

[43] V. Gervasi and D. Zowghi, "Reasoning about inconsistencies in natural language requirements," *TOSEM*, vol. 14, no. 3, pp. 277–330, Jul. 2005.

[44] A. Van Lamsweerde, R. Darimont, and E. Letier, "Managing conflicts in goal-driven requirements engineering," *TSE*, vol. 24, no. 11, pp. 908–926, 1998.

[45] S. Easterbrook and B. Nuseibeh, "Using viewpoints for inconsistency management," *SEJ*, vol. 11, no. 1, pp. 31–43, 1996.

[46] G. Perrouin, E. Brottier, B. Baudry, and Y. Le Traon, "Composing models for detecting inconsistencies: A requirements engineering perspective," in *REFSQ'09*. Springer, 2009, pp. 89–103.

[47] A. Sardinha, R. Chitchyan, N. Weston, P. Greenwood, and A. Rashid, "Ea-analyzer: automating conflict detection in a large set of textual aspect-oriented requirements," *ASE*, vol. 20, no. 1, pp. 111–135, 2013.

[48] M. G. Pitts and G. J. Browne, "Stopping behavior of systems analysts during information requirements elicitation," *JMIS*, vol. 21, no. 1, pp. 203–226, 2004.

[49] B. Boehm, "Verifying and validating software requirements and design specifications," *Software, IEEE*, vol. 1, no. 1, pp. 75–88, 1984.

[50] D. Zowghi and V. Gervasi, "The Three Cs of Requirements: Consistency, Completeness, and Correctness," in *REFSQ'02*, 2002, pp. 155–164.

[51] S. España, N. Condori-Fernandez, A. Gonzalez, and O. Pastor, "Evaluating the completeness and granularity of functional requirements specifications: A controlled experiment," in *RE '09*, 2009, pp. 161–170.

[52] C. B. Haley, R. Laney, J. D. Moffett, and B. Nuseibeh, "Security requirements engineering: A framework for representation and analysis," *TSE*, vol. 34, no. 1, pp. 133–153, 2008.

[53] V. N. L. Franqueira, T. T. Tun, Y. Yu, R. Wieringa, and B. Nuseibeh, "Risk and argument: A risk-based argumentation method for practical security," in *RE'11*, 2011, pp. 239–248.

[54] I. Mirbel and S. Villata, "Enhancing Goal-based Requirements Consistency : an Argumentation-based Approach," in *Int. Work. Comput. Log. Multi-Agent Syst.*, 2012, pp. 110–127.

[55] S. Ingolfo, A. Siena, J. Mylopoulos, A. Susi, and A. Perini, "Arguing regulatory compliance of software requirements," *DKE*, vol. 87, pp. 279–296, 2013.

[56] I. J. Jureta, J. Mylopoulos, and S. Faulkner, "Analysis of multi-party agreement in requirements validation," *RE'16*, no. November 2016, pp. 57–66, 2009.

[57] E. Bagheri and F. Ensan, "Consolidating Multiple Requirement Specifications Through Argumentation," *ACM SAC*, pp. 659–666, 2011.