Mining Incoherent Requirements in Technical Specifications

Patrick Saint Dizier

Mining Incoherent Requirements in Technical Specifications

Patrick Saint-Dizier(✉)

IRIT - CNRS, Toulouse Cedex, France
stdizier@irit.fr

Abstract. Requirements are designed to specify the features of systems. Even for a simple system, several thousands of requirements produced by different authors are often needed. It is then frequent to observe overlap and incoherence problems. In this paper, we propose a method to construct a corpus of various types of incoherences and a categorization that leads to the definition of patterns to mine incoherent requirements. We focus in this contribution on incoherences (1) which can be detected solely from linguistic factors and (2) which concern pairs of requirements. These represent about 60% of the different types of incoherences; the other types require extensive domain knowledge and reasoning.

Keywords: Linguistics of requirements · Incoherence analysis

1 Motivations and Objectives

Requirements (or business rules) form a specific class of documents with specific functions: they are designed to describe a task, a regulation or any kind of situation in a sound way. A requirement may be composed of a conclusion alone or be associated with one or more supports that justify it (these supports are usually called the rationale' of the requirement). In some specifications, warnings describe the consequences of not following requirements, while advice describe optional requirements that may improve e.g. the result of a task. Both warnings and advice are specific forms of arguments [1]. Requirements state in a declarative way e.g. the features a product should have, the way an organization should be managed via rules, etc. Even for a small product, several thousands of requirements are often necessary.

In spite of numerous authoring guidelines, unclear, ambiguous, incomplete or poorly written requirements are relatively frequent. This results in a significant waste of time to understand a specification, in difficulties to update, trace and re-use these specifications, and, more importantly, in risks of misconceptions by users or manufacturers leading to incorrect realizations or exposure to health or ecological risks. Controlling how requirements are written, independently of whether guidelines are followed or not, is a high and fast return-on-investment activity. For example, in the maintenance sector, poorly written requirements can entail extra costs up to 80% of the initial product development costs.

DOI: 10.1007/978-3-319-59569-6 8

The different protagonists involved in requirement production is a source of mismatches, inconsistencies and redundancies: stakeholders, technical writers, validators, users and manufacturers may all play different roles and have different views on the requirements. This is developed in e.g. [4,13–15,17]. Most industrial sectors have now defined authoring recommendations, methods and tools (e.g. Doors), to elaborate, structure and write requirements of various kinds, e.g. for easier traceability, control and update. However, our experience shows that those recommendations, in spite of the existing tools, cannot strictly be observed in particular for very large sets of requirements. Authoring tools have emerged two decades ago, e.g. one of the first investigated at Boeing [16], then at IBM with Acrolink and Doors, and, more recently, Attempto [2] that has some reasoning capabilities. [3] among many others developed a number of useful methodological elements for authoring technical documents. A synthesis of controlled language principles and tools is presented in [7,8]. Most of these principles and tools have been customized to requirement authoring, based on general purpose or domain-dependent norms (e.g. INCOSE, IREB or ISO26262).

Several road-maps on requirement elicitation and writing (e.g. [17]) show the importance of having consistent and complete sets of requirements, and rank it as a priority. However, no concrete solution so far, to the best of our knowledge, has been developed to characterize the coherence problem. Projects such as [6] aim at finding contradictions between lexico-syntactic patterns in Japanese, including spatial and causal relations. This is however quite different from the problem that is addressed here, since inconsistencies may have very diverse forms (Sect. 4). One of our main aim is indeed to characterize these forms w.r.t. requirement authoring principles. In [9], the limits of textual entailment as a method to detect inconsistencies is shown, corpora is developed from which a typology of contradictions is constructed. The need of a very fine granularity in the data is advocated to avoid errors.

Finding out incoherences in specifications is recognized by requirement authors to be a crucial and a very hard problem. Of interest is the site: www.semanticsimilarity.org that offers several tools for measuring similarities in texts. The use of SAT solvers (satisfiability solvers, running efficiently and in a sound way on very large sets of propositions) can be foreseen but this means translating requirements into propositional logic or into Horn clauses. Both of these translations fail to capture several facets of requirements, in particular complex VPs, modals and the discourse structure. Our goal is rather to develop specific linguistic patterns that can identify incoherences between requirements. These patterns could be viewed, possibly, as instantiated SAT templates.

In this article, we first show how a corpus of incoherent requirements can be constructed and annotated. This corpus leads us to develop a categorization of inconsistencies based on linguistic considerations. This contribution opens new perspectives (1) on new forms of incoherence mining in texts and (2) on mining incoherent requirements and arguments more generally. The construction of this corpus is extremely challenging: large volumes of requirements are necessary, that experts are not ready to share. Furthermore, incoherent requirements are in general not adjacent in a text and involve various textual and discourse structures.

2 Construction of a Corpus of Incoherent Requirements

Constructing a corpus of incoherent requirements is very challenging, it is difficult (1) to get real and substantial specifications from companies and (2) to extract pairs of incoherent requirements which are relatively sparse and may appear in remote sections or chapters of a specification.

Incoherence between two requirements may be partial: they may include divergences without being completely logically opposed. Next, incoherence may be visible linguistically, or may require domain knowledge and inferences to be detected and characterized. We focus in this research on incoherences which can be detected from a linguistic and general semantic analysis: these incoherences are domain and style independent and therefore mining them is simpler and probably re-usable over domains. Finally, we focus on incoherences between pairs of arguments, leaving aside incoherences which may arise from larger sets of requirements. It would obviously be more interesting and useful to consider more complex configurations: requirement compared to a sequence of requirements, or chains of requirements, but this involves more language complexity, e.g. the taking into account of discourse, co-text, titles, etc. A gradual approach to such complex forms, based on use-cases will probably be developed in the future.

2.1 Corpus Compilation Method

Our analysis of requirement incoherences is based on a corpus of requirements coming from 5 companies in three different critical industrial sectors (energy, transportation and telecommunications). Companies have requested to be anonymous; named entities (e.g. equipment and process names) in these texts have been replaced by meaningless constants. Specification documents need to be relatively long (above 100 pages) to allow the detection of various forms of incoherences; short documents are easier to manually control and have much less incoherence problems. Most specification documents, even for a simple equipment can be very long, which motivates our project.

Our documents are in French or in English. Our corpus contains about 1200 pages extracted from 7 documents, where only the requirement parts of these documents have been kept (leaving aside e.g. introductions, summaries, contexts and definitions, but also diagrams and formula). The requirement part of each document is between 150 and 200 pages long, i.e. between 3000 and 4000 requirements in each document. This is not very large, but seems to be sufficient to carry out this first analysis. A total of about 26000 requirements are considered. Most documents do not follow very accurately the norms for writing requirements, nor do they follow the guidelines imposed by their company. Reasons are not investigated here.

The main features considered to validate our corpus are the following, where diversity of form and contents captures the main linguistic features of requirements:

– requirements correspond to various professional activities, and have been written by different types of authors, over a relatively long time span (about a year),
– requirements correspond to different conceptual levels, from abstract considerations to technical specifications,
– requirements have been validated and are judged to be in a relatively 'final' state,
– requirements follow various kinds of authoring norms imposed by companies, including predefined patterns (boilerplates).

2.2 Extraction of Incoherent Requirements

It is almost impossible to manually extract incoherent arguments over large texts since this means memorizing with great precision several thousands of requirements. Specification authors can intuitively, via their knowledge of the domain, identify a very limited number of incoherences when proofreading texts, but they know that there are more incoherences, and that these may have important consequences.

The hypothesis we consider to mine incoherent requirements is that they deal with the same precise topic, but they differ on some significant element(s) which leads to the incoherence. Since requirements should a priori follow strict authoring guidelines (no synonyms, limited syntactic forms), dealing with the same precise topic means that two requirements which are potentially incoherent have a relatively similar syntactic structure and a large number of similar words, but differ on a few words. This can be illustrated by the two following requirements found in two different chapters of a specification, concerning the same software:

REQ12-7 A minimum of 10 simultaneous requests must be allowed.
REQ34-5 At least 20 simultaneous requests must be allowed.

Our strategy to mine potentially incoherent arguments is the following:

(1) In a specification, identify requirements among other types of statements such as definitions. Requirements are often labeled. If this is not the case, a grammar is used that identifies requirements based on a few specific clues such as the use of modals [5].
(2) Mine pairs of requirements which are potentially incoherent based on the two metrics developed below.
(3) Process the discourse structure of mined requirements. This is realized via the TextCoop discourse processing platform [11]. Of much interest are conditionals, circumstances, goals, purposes, illustrations and restatements. These structures are tagged in each requirement. For example: *<circumstance> when the temperature is below* 2°C, *</circumstance> the engines must not be switched off <purpose> to avoid any icing </purpose >*.
(4) Finally, manually inspect the results to identify, for each requirement pair that has been mined, if they are incoherent or not.

Mining pairs of potentially incoherent requirements is based on two metrics:

(1) a similarity metrics, since incoherent requirements deal with the same precise topic and therefore have very close linguistic contents, and
(2) a dissimilarity metrics to characterize differences.

Since, at this stage, the form of incoherent requirement pairs is unknown, we developed two metrics with general purpose constraints in order to mine a large diversity of situations, the noise being discarded manually in a later stage (step 4 above).

(a) The Similarity Metrics. Requirements follow style guidelines: requirements are short (in principle they must be smaller than 20 words, but in practice 30 words are frequently encountered), synonym terms are not allowed, the syntactic structure is very regular (passives are not allowed), negation must be avoided, adjectives and adverbs are limited to a few, fuzzy terms are not allowed, etc. Therefore, a similar content can be characterized, in a first experiment, by a large set of similar words shared by two requirements. Only the words that have a topical content are included in the metrics, to form the *content requirement signature* S. They are: nouns, proper nouns, numbers, verbs, symbol names, and boolean or scalar adjectives. Their identification is based on WordNet resources. The other terms, which mostly play a grammatical role, are not used in the metrics, they include: prepositions, adverbs, determiners, modals, auxiliaries if they are not the main verb, negation, connectors, coordination and pronouns. They may play a role in the incoherence analysis, but do not convey the topic of the requirement.

For example, in *the maximum length of an imaging segment must be 300 km*, the words that are considered in the metrics are: maximum, length, imaging, segment, be, 300 km. These form the *content requirement signature* of that requirement, composed of 7 terms. The similarity metrics considers the *content requirement signatures* S_1 and S_2 of two requirements R_1 and R_2 and computes their intersection I and union U, considering the noninflected forms of words. The similarity rate is: I/U.

Requirements have in general between a total of 6 and 40 words, with an average size of 22 words. In this initial experiment, after some tuning from a small sample of requirements, it turns out that two requirements are in a similarity relation, if:

- between a total of 6 and 12 words long, the similarity rate is greater or equal to 80%,
- between a total of 13 and 22 words long, the similarity rate is greater or equal to 70%,
- above a total of 22 words long, similarity rate is greater or equal to 65%.

Indeed longer requirements often include peripheral information which means that the threshold for the similarity weighted rate should be lower. This 3 level tuning can still be slightly improved, depending on the authoring style of the requirements, the domain via its terminology and the conceptual complexity.

(b) The Dissimilarity Metrics. The dissimilarity metrics considers those terms which are different independently of the discourse structure. It parallels the similarity metrics, searching for terms which introduce a clear difference. Intuitively, differences may be characterized by the main following situations:

- use of different values for the same statement, with a difference of at least 10% to be significant,
- use of different arithmetical operators and constraints,
- use of different named entities,
- use of antonym prepositions or connectors, e.g.:*before/after, neutral/after,* etc.
- use of contrasted modals: e.g.: *must/recommended,*
- presence of typical structures such as manner or temporal adverbs in a requirement and not in the other,
- presence of two main clauses (two events) in one requirement and a single main clause in the other (single event),
- presence of a conditional or a circumstance structure in one the of the requirements and not in the other one.

Provided that two requirements have been ranked as similar by the similarity metrics, the presence of at least one of these criteria in a pair of requirements entails that they may be incoherent. More than one criteria reinforces the risk of incoherence, however, there is no dissimilarity rate at this level.

These two metrics used jointly have been tested on our 7 texts separately since they deal with different topics. A total of 204 pairs of requirements have been mined as potentially incoherent. Then, from this corpus, via a manual analysis, 83 requirement pairs (on average 41%) that have been mined indeed show some form of incoherence. This is not a very high number of incoherent arguments: this confirms expert intuitions that incoherences are relatively sparse, but there are probably more incoherences. 18 pairs have been found in the same text section (i.e. among between about 80 to 150 requirements), while 65 pairs are more distant in the text and couldn't have probably been identified without a tool.

Even if it is small, this sample of 83 requirement pairs allows us to develop an analysis of incoherence among requirements, and then to develop relatively generic templates that go beyond these observations that should be able to mine a higher number of incoherent requirement pairs. Finally, a side effect of this analysis is to also mine coherent duplicates or very closely related requirements which could be eliminated. Via the similarity metrics used alone, with a threshold of 90%, our corpus contains at least 368 requirements which are very closely related and could be analyzed as duplicates.

3 Annotating Incoherence in Requirements

The next stage of our analysis is to identify more precisely the various forms incoherence takes in requirements. For that purpose, we defined a set of annotations. The discourse processing used to mine requirement pairs provides a first

level of annotations as illustrated in the previous section. Annotations are made manually by the author, some confirmation were given by requirement authors, but their availability is often very limited.

The annotations we add are those that characterize the string of words which differ between the elements of a pair and the nature of the difference(s). This task allows to categorize errors (Sect. 4) and to define templates to mine incoherent arguments in specifications. Incoherences are specified informally as an attribute of the main tag <incoherence>. Here are a few examples that illustrate the method and the difficulties. Examples 1 and 2 are relatively straightforward whereas Examples 3 and 4 are much more complex to annotate and to characterize. Differences are between <diff> tags, <req> identifies a requirement, and <event> tags the events in requirements that contain two or more explicit events.

Example 1. <incoherence type = "numerical variation" incoherence = "300 ≠ 100">
<req> the maximum length of an imaging segment is <diff> 300 </diff> km. </req>
<req> the maximum length of an imaging segment is <diff> 100 </diff> km. </req>
</incoherence>

Example 2. <incoherence type = "arithmetical expression divergence" incoherence = "(<30) ≠ 50">
<req> In S, the probe X30 shall be preheated <diff> up to 30 </diff> degrees <circumstance> <diff> before doing</diff> E. </circumstance> </req>
<req> the probe X30 in S shall be preheated <diff> to 50 </diff> degrees <purpose> <diff> to realize </diff> E. </purpose></req> </incoherence>
In this example, while the arithmetical expression introduces the incoherence, the distinction between circumstance and purpose must be made flexible so that the synonymy between 'doing' and 'to realize' can be identified.

Example 3. English gloss of French example: *at flight level 30 in normal flight conditions, the maximal* 20° *flap extension speed is* 185 kts versus *at flight level 30 in normal flight conditions, extend flaps to* 20° *and then reduce speed below* 185 kts*:*
<incoherence type = "event synchronization with mismatches" incoherence = "(< kts) ≠ unknown">
<req> <circumstance> A FL 30 ASL et dans les conditions normales </circumstance>, la vitesse <diff> maximale de </diff> sortie des volets à 20° <diff> est </diff> 185 kts. </req>
<req> <circumstance> A FL 30 ASL et dans les conditions normales </circumstance>, <event> sortir les volets à 20° </event> puis <event> <diff> réduire la vitesse en dessous de </diff> 185 kts. </event> </req> </incoherence>
In this example, a general rule (first requirement) is contrasted with a requirement that describes a procedure composed of two events. In the second requirement, the temporal connector 'puis' (then) suggests that the constraint stated in the first requirement is split into two consecutive parts, with the risk that the

constraint is not met as it should, e.g. flaps are extended at an unknown speed, which is then reduced to 185 kts.

Example 4. <incoherence type = "incompatibility between events" incoherence = "(stop every 24 h) ≠ (no stop during the entire update)">
<req> the system S administrator must make sure <diff> to stop the system S every 24 h <purpose> for maintenance. </purpose> </diff> </req>
<req> the system S administrator must make sure <diff> that the update of system S database is not interrupted. </diff></req> </incoherence>.

In this example, a potential incoherence may arise if the system is stopped for maintenance while its associated database is being updated. The terms 'stop' and 'not interrupted' are opposed and may create the incoherence.

These annotations remain informal, in particular for the 'type' attribute. Incoherences are very diverse and do not lend themselves easily to a simple characterization. The goal is to have a first level of analysis, before implementing templates that would mine incoherent pairs of requirements. The examples given above are composed of independent statements. Connecting these statements via e.g. conditional statements could improve their global coherence.

4 A Preliminary Categorization of Incoherence in Requirements

From our sample of 83 requirement pairs which are incoherent, a preliminary categorization of incoherence types can be elaborated on a linguistic and conceptual basis. This categorization shows that incoherence has multiple facets, and that some pairs that have been mined may be a symptom of a form of incoherence. This categorization leads to the definition of templates and lexical resources to mine pairs of incoherent requirements. The examples below are direct corpus samples (translated into English for French examples). They are numbered Ri a/b where i numbers examples and a and b are the 2 elements of an incoherent pair.

4.1 Partial or Total Incompatibilities Between Expressions

The most direct and simple type of incoherence is composed of 'local' divergences:

- incompatible numerical values or arithmetical expressions, with more than 10% difference:
  R1a- Make sure to preheat the probe X30 to a maximum of 30°.
  R1b- The probe X30 must be preheated at at least 50°.
  R'1a- The maximum length of an imaging segment is 300 km.
  R'1b- The minimum length of an imaging segment is 30 km.

– incompatibility between various types of temporal, spatial or instrumental restrictions expressed as verb complements:
R2a- Service D must be available only from the screen.
R2b- Service D must be available in any configuration.
A domain ontology is necessary to detect most of these incompatibilities. However, the fact that a different term is used can be identified as a potential incompatibility source, since in technical writing synonyms are not allowed.
– incompatible temporal structures or temporal organization between events:
R3a- Update data marking when transferring data.
R3b- Update data marking before transferring data.
– modal variation, involving e.g. a different critical level:
R4b- It is recommended to stop the engine before E.
R4b- The engine shall be stopped before E.
– adverbial incompatibilities or discrepancies in particular in manners: *carefully/quickly*, etc.
– some specific quantification aspects such as the difference between *every/each, most/all.*

Among these various forms of incoherences, the three first cases are the most frequent. Adverbial uses and quantification are less frequent: they are strongly controlled in requirement authoring because they often introduce underspecified or fuzzy expressions. Our corpus includes 33 cases that fall in this category, among which 24 belong to the three first cases.

4.2 Incompatible Events

In this category fall pairs of requirements that describe events which show some form of incoherence. Such incoherences often require some domain dependent knowledge, but a number of them remain at a 'surface' level and can be detected solely on a linguistic basis. Examples 3 and 4 in Sect. 3 are typical incoherent statements:

R5a- The system S must be stopped every 24 h *for maintenance.* versus
R5b- The update of the database via the system S must not be interrupted.
is a type of situation that is difficult to predict besides the antonym *stop/not interrupted.*
R6a- the maximum 20° *flap extension speed is* 185 kts. versus
R6b- extend flaps to 20° *and then slow down to* 185 kts.
is a typical business error that any domain expert should be able to avoid. R6a is a general rule which should be preferred to R6b which is a procedural statement.

4.3 Contextual Incompatibilities

This category includes various types of discourse structures (e.g. condition, circumstance, goal, illustration, etc.) associated with the main body of two similar requirements which could induce forms of incompatibility:

R7a- during the project specification phase, both on-line and off-line access must be available.
R7b- during the project implementation, both on-line and off-line access must be available.
R8a- a rule includes facts and its critical.
R8b- a rule includes facts and a description.

R7 illustrates a case where the context (circumstance) is different but expects the same behavior. One may expect a negation in one of the two main propositions 'must not be available' to motivate these two requirements or may want to merge the two circumstances to avoid any doubt. R8 shows two enumerations which are different and should probably be adjusted. R8 is a frequent case where enumerations are either incomplete (not recommended in requirement guidelines), or manipulate objects at different levels of abstraction.

4.4 Terminological Variations and Other Discrepancies

In this category, requirements which largely overlap are considered. Their slight differences may however be symptomatic of an partial inconsistency. These cases are relatively frequent and typical of documents produced either by several authors or over a long time span (where e.g. equipment names may have changed). Typical examples are:

R8a- the up-link frequency, from earth to space, must be in the s band.
R8b- the down-link frequency, from space to earth, must be in the s band.
R9a- those tests aim at checking the rf compatibility of the ground stations used during the mission and the tm/tc equipment on board.
R9b- those tests aim at checking the rf compatibility of the ground stations used during the mission and the on board equipment.
R10 is somewhat ambiguous and contains implicit knowledge which makes the identification of the incoherence slightly more challenging:
R10a- the upper attachment limit must not exceed 25 GB.
R10b- It must be possible to specify a maximum limit for the storage capacity of an attachment.

Indeed, either the limit is set to 256 GB or it can be specified, but R10b may be understood as saying that it can be specified but always below 256 GB.

4.5 Category Synthesis

The distribution observed on our corpus is the following, it remains indicative because of the small size of our corpus:

The implementation of these categories could possibly produce slightly different results. Most of the incoherences in these categories can be detected on the basis of linguistic analysis and general purpose lexical semantics data, in particular antonyms e.g. for prepositions, temporal connectors, arithmetical operators.

Category	nb of occurrences and rate
(1) incompatibilities between expressions	33, about 40%
(2) incompatible events	11, about 13%
(3) contextual incompatibilities	27, about 33%
(4) variations between requirements	12, about 14%

5 Analysis of Errors in the Incoherence Analysis

The main challenge is now the definition of templates to mine incoherent pairs of requirements, with their associated linguistic resources. The 41% accuracy of the current mining is not sufficient: a rate of 75% is the minimum acceptable accuracy for requirement authors. This means refining both metrics and analyzing the reasons of the noise to reduce it. For that purpose, a first step is to analyze why some requirement pairs have been incorrectly recognized as incoherent. The main reasons are the following:

(1) Description of complex cases via several coherent requirements: requirements are often detailed descriptions of a precise case within a given context. For complex cases, a set of closely related requirements may be produced where a few terms, dealing with the different cases, may be different, yet requirements remain coherent.

(2) Different forms for a similar content: two requirements may deal with the same point using slightly different expression means, e.g. for values. For example, values+units may be different, intervals or arithmetical constraints may be used instead of a single value, but these expressions remain globally equivalent. For example, these two requirements are equivalent according to the similarity metrics: R11a- *the A320 neo optimal cruise altitude is FL380 in normal operating conditions* versus R11b- *the A320 neo optimal cruise altitude is between FL 360 and 380, depending on weather conditions.* R11b is just more precise.

(3) Use of more generic terms: it is also frequent that two requirements differ only in a general purpose term or a business term where one is more generic than the other, or with a language level that is different. Detecting this situation requires a domain ontology.

(4) Two or more differences between two requirements: when two requirements have more than two groups of terms which are different, it turns out that in most cases they deal with different aspects of a given topic. The dissimilarity analysis should be revised so that the case of several observed differences is constrained, for example manner or temporal adverbs and different values or arithmetical expressions as advocated in 2.2(b) can co-exist, but not two groups of different values.

(5) Presence of negative terms: the negation, although not recommended in technical writing, or negatively oriented terms (verbs, adjectives) may appear in one requirement and not in the other, but these requirements are in fact similar to a large extent. For example R12a- *Acid A must not be thrown in*

any standard garbage versus R12b- *Acid A must be thrown in a dedicated garbage.*

(6) Influence of the co-text: requirements dealing with a given activity are often grouped under a title, subtitle, an enumeration or in a chart. Two arguments that belong to two different groups may seem to be incoherent, but if the context, given by the title or by the enumeration introduction head is different, then these two requirements may deal with different cases as in (1) and may not be incoherent. Co-text aspects are crucial in incoherence, but quite difficult to take into account because the link between a co-text and the requirement needs to be analyzed at the discourse level.

In terms of silence, two requirements have not been detected as incoherent because of:

(7) implicit elements: this is the case in the pair R10 (detected in a different way) where the implicit 'for the storage capacity' expression is unexpressed in pair a. The same situation is observed with pronouns, although their use should be very limited in technical authoring. Missing information prevents the similarity metrics from mining requirements dealing with the same precise topic.

(8) an external context: Similarly to (6) above, but in the other direction, two requirements may be exactly identical but incoherent if the sections in which they appear have titles which are opposed in some way. The incoherence may then be 'external' to the requirements. However the case structure evoked in (1) may also be considered here.

More observations are probably necessary before a model and an implementation for the detection of incoherent requirement pairs can be carried out. However, a few simple remarks can be made at this stage. Category 1 in Sect. 4 should be relatively simple to implement, with a base of antonyms, various forms of divergence expression, and a limited syntactic analysis for verb complements. Some elements that fall in category 3 should also be relatively simple to implement, after a discourse analysis of the requirements, but this category may include more complex cases than those given as illustrations. Categories 2 and 4 are much more challenging and probably the most interesting ones from a scientific point of view.

The similarity metrics clearly needs to be refined: a simple word to word match and count is obviously not accurate enough, even if it allowed us to investigate the incoherence problem from scratch. Several problems such as pronominal references, the use of more generic terms, and implicit terms, and ellipsis must be taken into account in some way.

6 Conclusion and Perspectives

We have presented in this paper the construction of a corpus of incoherent requirements and a preliminary categorization. We have restricted ourselves to incoherences which can be detected on a linguistic basis. Mining incoherences is really challenging as shown by the analysis provided for each category. Modeling and implementation is ongoing, but requires more data.

This contribution opens new perspectives (1) on new forms of incoherence in texts and (2) on mining incoherent requirements, and arguments more generally. Our corpus examples show that incoherence may take a large diversity of forms.

A major difficulty is evaluation. Since it is not possible to have a test corpus where incoherent requirements have been identified manually, it is only possible to evaluate noise, but not silence. Both dimensions are important to evaluate the accuracy and also the linguistic adequacy and soundness of templates. In addition, requirement authors certainly do not want to be bothered by getting alerts about pairs of requirements which are not incoherent, they also wish to have an estimate of what the system found.

References

1. Fontan, L., Saint-Dizier, P.: Analyzing the explanation structure of procedural texts: dealing with advices and warnings. In: STEP Conference, Venice (2008)
2. Fuchs, N.E.: First-order reasoning for attempto controlled English. In: Rosner, M., Fuchs, N.E. (eds.) CNL 2010. LNCS, vol. 7175, pp. 73–94. Springer, Heidelberg (2012). doi:10.1007/978-3-642-31175-8_5
3. Grady, J.O.: System Requirements Analysis. Academic Press, Orlando (2006)
4. Hull, E., Jackson, K., Dick, J.: Requirements Engineering. Springer, London (2011)
5. Kang, J., Saint-Dizier, P.: Discourse structure analysis for requirement mining. Int. J. Knowl. Content Dev. Technol. **3**(2), 43–65 (2013)
6. Kloetzer, J., De Saeger, S.: Two-stage method for large-scale acquisition of contradiction pattern pairs using entailment. In: Proceedings EMNLP 2013 (2013)
7. Kuhn, T.: A principled approach to grammars for controlled natural languages and predictive editors. J. Logic Lang. Inf. **22**(1), 33–70 (2013)
8. Kuhn, T.: A survey and classification of controlled natural languages. Comput. Linguist. **40**(1), 121–170 (2014)
9. De Marneffe, M.C., Rafferty, A.N.: Manning CD finding contradictions in text. In: ACL-HLT 2008 (2008)
10. O'Brien, S.: Controlling controlled English–an analysis of several controlled language rule sets. School of Applied Language and Inter-cultural Studies, Dublin City University (2003)
11. Saint-Dizier, P.: Processing natural language arguments with the <TextCoop> platform. J. Argumentation Comput. **3**(1), 49–82 (2012)
12. Saito, M., Yamamoto, K., Sekine, S.: Using phrasal patterns to identify discourse relations. In: ACL 2006 (2006)
13. Schriver, K.A.: Evaluating text quality: the continuum from text-focused to reader-focused methods. IEEE Trans. Prof. Commun. **32**, 238–255 (1989)
14. Unwalla, M.: AECMA simplified English (2004). http://www.techscribe.co.uk/ta/aecma-simplified-english.pdf
15. Weiss, E.H.: How to Write Usable User Documentation. The Oryx Press, Westport (1991)
16. Wojcik, R.H., James, E.H.: Controlled languages in Industry, in survey of the state of the art in human language technology, Chap. 7 (1996). http://www.lt-world.org/hlt-survey/ltw-chapter7-6.pdf
17. Wyner, A., Angelov, K., Barzdins, G., Damljanovic, D., Davis, B., Fuchs, N., Hover, S., Jones, K., Kaljurand, K., Kuhn, T., Luts, M., Pool, J., Rosner, M., Schwitter, R., Sowa, J.: Properties and prospects. http://wyner.info/research/Papers/CNLP&P.pdf