An ISO/IEC 24744-Derived Modelling Language for Discourse Analysis

Patricia Martin-Rodilla, Cesar Gonzalez-Perez
Institute of Heritage Sciences
Spanish National Research Council
Santiago de Compostela, Spain
{patricia.martin-rodilla, cesar.gonzalez-perez}@incipit.csic.es

Abstract—Most of the information that is used as input for the development of information systems is originally produced in non-structured forms, such as verbal communications or free-style text documents. Some examples are requirements specifications or documents associated to translation of contents. In cases like these, there is a need to support the structuring and extraction of the underlying semantic relations embedded in the text, in order to fully understand and post-process it. There are models for the analysis of textual information, such as information retrieval solutions or topic maps. However, these models do not offer an integral, flexible and reusable approach that can assist in giving structure and semantics to the information within a standard framework. Discourse analysis techniques, in contrast, identify semantic relations (such as authors’ intentionality or possible dependencies between text elements) in the textual information according to well-established linguistic patterns. Based on these techniques, this paper presents a modelling language based on the ISO/IEC 24744 metamodel that is capable of representing pieces of textual information in a highly structured manner, describing the semantic relations in the associated discourse. In addition, the paper shows an application of the proposed language to the domain of requirements engineering, illustrating the benefits of the application of the suggested approach as well as its possibilities in other textual domains.

Keywords—discourse analysis; metamodelling; modelling language; ISO/IEC 24744; requirements engineering

I. INTRODUCTION

Most of the information that is used as input for the development of information systems is originally produced in non-structured forms, such as verbal communications or documents written in a free style. Some examples include requirements specification documents or documents produced to translate contents to different languages.

The non-structured form of these products arises naturally from the way people communicate, but hinders an agile analysis of the information, since the involved semantics can only be understood by humans rather than computers. In addition, non-structured information needs large human efforts to be re-structured and characterized before further processing, through ad hoc databases or semantically linked systems. Hence, the necessity of a better conceptualization and structuring of textual information has been detected. Compliance to requirements, meaningful integration of legacy data into new software systems, or the generation of adequate Textual Data Mining (TDM) sources are examples of situations that motivate this necessity [1-3].

There are two major approaches to fulfill this need: an information retrieval approach and an ad-hoc domain modelling approach.

On the one hand, the information retrieval approach has developed a large corpus to analyze textual information in an automatic process, with heuristic and probabilistic approximations as well as semantic approximations [4]. Heuristic and probabilistic approximations focus on the extraction of information from textual sources in a quantitative level [5, 6], for instance counting occurrences in a specific text, or looking at frequency indicators or automatic indexation candidates. Thus, this approach allows us to obtain quantitative information, but in most cases is unable to provide information about the semantic relations between elements in the text. More semantically-oriented approximations allow us to analyze textual sources based on thesauri or topic maps [7, 8], permitting the extraction of semantic information about the information structure and semantic relations inside it. It is possible, for instance, to detect equivalence relations or hierarchical relations between elements in the text. These semantic approaches have been successfully applied, for instance, to find the common lexeme in a family of words, in order to analyze all these words as a group [9, 10]. However, the semantic relations that can be identified by using these techniques are not as powerful as those that can be obtained by using other techniques with a heavy linguistic background, such as discourse analysis, which allow the identification of not only equivalence or hierarchical relations, but also causal, consequence or exemplification relations in the text. In addition to the explained limitations of both cases (heuristic and probabilistic solutions, and semantic approaches), information retrieval approaches need an increased abstraction level in the techniques to become in a real domain-independent solutions and be able to extract semantic knowledge from free-style documents.

On the other hand, ad-hoc domain modelling approaches have experienced a large growth in recent years, specially related to the application of textual analysis to fields with high structuring needs such as biomedicine [11, 12]. These approaches have been successfully applied to add structure to non-structured information, but work only inside the context of a well-defined domain. Once this limitation is assumed, a very accurate conceptual model can be created that captures the semantic relations between pieces of text. However, this
conceptual model needs to be created ad hoc for each application: each new text analysis involves a new conceptual model creation process. For these reasons, it is not possible to attain a high degree of standardization.

These identified limitations in the structuring and semantic comprehension of free-style texts that are used as input for the development of information systems motivates our proposal: a modelling language capable of representing pieces of textual information in a highly structured manner, describing rich semantic relations based on discourse analysis techniques. The language is based on the ISO/IEC 24744 standard in order to facilitate the connection of the language to other method elements and its integration into full-fledged information systems methodologies. We need to emphasize that, albeit this language may be applied through specific tools for efficiency and ease of use, the proposal presented here encompasses the language only rather than a full technology stack, very much like the UML [29] specification proposes a language without describing any tools that may be developed to implement it.

The paper is structured as follows: Section II describes background work on discourse analysis and the formalization of textual discourse. Section III introduces the ISO/IEC 24744 standard. Section IV shows the proposed modelling language. Section V illustrates how the language works through a case study in the requirements engineering domain. Section VI describes other approaches to structure and understand textual information different to discourse analysis, and compares them to the proposal presented here. Finally, Section VII describes the application possibilities of the proposed modelling language to other textual domains, offers some conclusions and outlines future work.

II. DISCOURSE ANALYSIS

As outlined above, our proposal involves the use of discourse analysis techniques to structure free-text information in information systems development processes and to extract semantic relations from it. This means that, although the semantics of the text are already contained in it, the IS development process needs that these semantics are made much more explicit and unambiguous. This section describes the main concepts behind discourse analysis and its applications.

Discourse analysis constitutes a vast field in linguistics that emerged from the necessity to discover meaning in terms of the narrative elements presented in a discourse. According to Zelling Harris, who originally defined the term, discourse analysis is "a method for the analysis of the connected speech or writing for continuing descriptive linguistics beyond the limit of a single sentence at a time and for correlating culture and language" [13]. Apart from other applications, discourse analysis techniques are used to study verbal communications and textual documents, focusing in the organization of the language, above or under sentence or paragraph level.

As a result of a discourse analysis session, discourse elements are identified; these discourse elements provide information about the structure of the narrative and the reasoning or intention of the speaker or writer. There are several techniques to analyze a discourse, which are used depending on the domain involved or the focus of the analysis: the structure of the clauses, the function of particular discourse elements, or the representation of texts [14].

Currently, discourse analysis is successfully applied to different fields in different contexts, with different degrees of automation. Some examples are biomedicine [15], with applications focused on extracting analytical information or legal texts [16], with applications focused on identifying consequence relations between elements in the texts in order to detect the possible consequences of a specific legal action. In all cases the analysis of the discourse helps to understand the reasoning modes in the domain studied.

Due to the potential of discourse analysis techniques to identify the semantic relations "hidden" in texts, we consider them as a suitable and agile basis to structure and to extract semantic relations in free-style textual information, including textual information produced in the IS development process. There are specific applications that allow us to analyze a set of documents, avoiding the ad hoc discourse analysis per document. However, these applications are not formal and abstract enough as to be integrated in a complete IS development methodology. There are not formal enough because they are never expressed as a metamodel or similarly unambiguous device; they are not abstract enough because, as we have said, they often require ad hoc work. For these reasons, a general-purpose language that allows the application of discourse analysis to heterogeneous areas is necessary, and which can work as part of an integral methodology for IS development.

In order to achieve this, we chose to follow the discourse analysis approach of Hobbs [17], because it allows the characterization of argumentation relations between clauses and it has been applied to IS development before [18, 19]. Also, this method has been used on narrative texts in different languages [20] and types of documents.

Hobbs’ approach describes an iterative process to analyze textual discourse. First of all, the major breaks in the text, typically paragraphs, are determined. This helps to maintain the semantic coherence of the text. The identification of breaks is an iteratively process, so the paragraphs obtained during the first pass will be recursively analyzed later to identify major breaks inside them until single clauses are reached.

Then, each pair of clauses is tagged as being related by one particular coherence relation. Hobbs’ method includes predefined coherence relations, including causal argumentations, relations of consequences, contrastive argumentations, exemplifications and generalization of arguments. The choice of coherence relation for any given pair of clauses is based on two aspects:

- The structure of the clauses involved, e.g. organization or presence of certain keywords, grammar connectors, etc. For instance, the presence of causal grammar connectors usually indicates that the causal coherence relation is a good candidate.
- The semantic references that the clauses make to domain elements. For instance, if both clauses describe one entity in the domain undergoing a change over time, this usually indicates that one of
the coherence relations involving temporal change is a good candidate.

As a third step, the components that make up the chosen coherence relation are identified in the associated clauses. Each particular coherence relation follows a given formal structure, and therefore a well-known set of components is expected. For example, every occasion coherence relation is expected to refer to one or more entities in two different temporal situations; the entity (or entities) and the two temporal situations must be identified.

Steps two and three are performed recursively from bottom to top in the text, starting from simple clauses and following the breakdown structure of the discourse “upwards”, until the largest sections have been analyzed. Thus, coherence relations are established not only between clauses but also between simple sentences, complex sentences and even whole paragraphs.

Finally, Hobbs’ approach includes a validation step. A common practice to validate the outcome of a discourse analysis is the assessment of the conclusions by experts in the domain.

The next section introduces the ISO/IEC 24744 standard, and explains the choice of this standard as a basis from which to derive a modelling language for discourse analysis.

III. THE ISO/IEC 24744 STANDARD

When a modelling language is designed, one must take into account not only its syntax and semantics, but also the methodological place that it is supposed to occupy. In other words, the language will be presumably used in situations that are part of a larger process, and where work artefacts (material or conceptual) are being generated, altered and used by people playing different roles. The language must, therefore, “plug into” this methodological setting so that, for example, whatever portion of reality the language is supposed to represent is guaranteed to be available at that point in time during the execution of the process; and, similarly, the models constructed with the language can be used by tasks and people downstream. The dangers of not thinking about this are illustrated in, e.g. [21].

One way to facilitate the methodological integration of a language is to express it as an instance of a standard metamodel for methodologies; this is exactly the approach that we have taken, and we have chosen the ISO/IEC 24744 standard [22, 23] as underpinning metamodel. ISO/IEC 24744 provides the basic conceptual constructs to define a modelling language, as well as to integrate it with the necessary process and people issues; this capability is absent from other metamodels such as OMG’s SPEM [24] as described in [25].

Fig. 1 shows a small fragment of ISO/IEC 24744 containing the most relevant classes for our purpose. The Language class in the bottom row represents a language, such as the one we are about to propose. The ModelKind class represents a particular kind of models that can potentially be used within a methodology, e.g. class models or use case models. The metamodel dictates that each model kind uses a well-known language. The Model class, in turn, represents a particular model that is constructed and/or used within a given enactment of a methodology, e.g. a particular class model or a particular use case model. Each model is of a specific kind, and this is captured in ISO/IEC 24744 through the concept of powertyping, which plays a crucial role. The concept of powertype was introduced in software engineering by [26] and applied to metamodeling by [27, 28]. In Fig. 1, the Model and ModelKind classes are depicted side-to-side because they make up a powertype pattern, i.e. ModelKind is a powertype of Model. In other words, the instances of ModelKind are also subtypes of Model. This means that any particular model kind that we might want to define using ISO/IEC 24744 would be represented by an object (an instance of ModelKind) as well as a class (a subtype of Model); in fact, this object and this class would represent the very same thing, namely the model kind we have in mind, and therefore they would make up a hybrid entity named a clabject, i.e. “class plus object”. For additional details on how powertype patterns and clabjects work, and what their role is in the definition of method components and methodology enactment, please see [27, 28]. ModelUnitKind and ModelUnit in Fig. 1 make up another powertype pattern, where ModelUnitKind represents a particular modelling primitive that can be part of a language, such as “class” or “attribute” in class modelling, or “use case” or “actor” in use
case modelling. *ModelUnit*, on the other hand, represents a particular occurrence of a model unit kind, for example, a particular class or attribute or use case.

Methodological integration of a language based on ISO/IEC 24744 is achieved through the semantics embedded in the metamodel; in our case, *ModelKind* in Fig. 1 is related to *TaskKind* (via *ActionKind*), so that the process that uses, creates or modifies a model can be readily captured through action kinds of different types (create, read-only, modify or delete). In turn, *TaskKind* is related to *ProducerKind* (via *WorkPerformanceKind*) so that the people and tools that participate in said actions can be easily expressed. These associations are described here briefly for the sake of simplicity, but are fully explained in [22].

The five classes illustrated in the bottom row of Fig. 1 will be used in the next section to create the proposed modelling language. The *Language* class, being stand-alone, will be instantiated in a regular way. The other four classes, however, will be used through powertype pattern instantiation as dictated by Clause 8.1.2 of the standard specification [22].

IV. PROPOSED MODELLING LANGUAGE

In this section we describe the details of the proposed modelling language. Fig. 2 shows how the ISO/IEC 24744 bottom row classes in Fig. 1 have been used to create the basis of this language. First of all, the *Language* class has been instantiated into an *L1* object with *Name* = “Discourse Language”. This object represents the language itself. Then, and as introduced in the previous section, the powertype patterns for *ModelKind* and *ModelUnitKind* have been instantiated to create the corresponding clabjects. In the case of *ModelKind*, a *DiscourseModel* class plus an *MK1* object have been created, which are, respectively, a subtype of *Model* and an instance of *ModelKind*. Both make up a clabject, depicted in Fig. 1 by a dashed ellipse; this clabject represents the specific kind of models that can be created by using the language that we have just defined, namely, discourse models. Similarly, a *DiscourseModelUnit* class plus a *MUK1* object have been created, which are, respectively, a subtype of *ModelUnit* and an instance of *ModelUnitKind*. Both make up a second clabject, which represents the modelling primitives that are part of said language and which can be used to compose models of said kind.

As explained in Section I, the final goal of the proposed language is to give structure to and extract semantic relations in the discourse analysis. More specifically, the major purpose of applying the proposed language and construct a model with it is to document the source discourse (a description of requirements, for example) in a highly structured fashion which can help developers to trace back and forth between the source text and the software artefacts (such as classes and attributes) that are later created. To achieve this purpose, the language must contain elements from three different relevant areas. First, we need to represent the narrative or discursive elements in the source text themselves, such as sentences, clauses and its aggregations. Secondly, the domain must be taken into account, since any discourse always refers to some part of reality, hence a domain. Following an object-oriented approach to modelling, this domain is structured around entities (i.e. objects) and values (i.e. slots) as well as types (i.e. classes) and features (i.e. attributes). These primitives are not meant to replace a complete modelling language such as UML [29] or ConML [30], but to work as a mapping point towards the modelling solution of choice. Having considered these two areas, the language will be able to model what fragment of the discourse refers to what specific entities; the *Reference* class acts precisely as the connector between these two.

Furthermore, a modelling language for discourse analysis needs to consider a third area, which, as we said in Section II, adds the most value. Following Hobbs’ approach, we model the different coherence relations that may occur in the discourse by using a class to represent each. Each particular kind of coherence relation uses references in its own particular way, and therefore each subclass of *CoherenceRelation* will be associated to *Reference* in a different manner.

The full details of three areas described are shown in Fig. 3. This figure also shows that Hobbs’ coherence relations have been arranged in two categories, which are implemented through two abstract classes: linguistic and formal relations. This classification is not present in Hobbs’ work, but has been added by us according to the degree of detail with which they refer to domain entities. On the one hand, we have been able to
describe formal relations in terms of what entities and values they refer to with a high level of detail. On the other hand, we have been unable to describe linguistic relations in terms of the details of what domain elements they refer to, but only in terms of what coarse-grained discourse elements are involved, such as sentences or clauses. An example of formal coherence relation is that of Parallelism, which is described in terms of a statement about two or more entities having similar values for the same features. An example of linguistic coherence relation is that of Explanation, which is described in terms of an element of discourse (such as a sentence or group of sentences) that explains why a previous element of discourse (which can even be implicit) has been said.

The arrangement of language elements in three separate areas, as well as the classification of coherence relations in two well defined subtypes, makes the language highly modular, and will facilitate changes in the future. For example, the addition of new coherence relations would be easy, not affecting the class structures under DomainElement or DiscourseElement. Similarly, a potential redesign of the DomainElement area would not affect the structure of the coherence relation classes.

Since a complete description of the metamodel for the ten coherence relations in Fig. 3 would need more space than is available, we will focus on two of them only. We have chosen OccasionRelation and ExemplificationRelation because they are quite common formal coherence relations, and because they will be useful when analyzing the case study presented in Section V.

A. Occasion Relation

Following Hobbs’ definition, an occasion relation is a coherence relation between two discourse elements that describe two events: “the first event sets up the occasion for the second. In both cases we let $S_I$ be the current clause and $S_0$ an immediately preceding segment. There are two cases:

- A change of state can be inferred from the assertion of $S_0$, whose final state can be inferred from $S_I$.
- A change of state can be inferred from the assertion of $S_I$, whose final state can be inferred from $S_0$.”

Elaborating on Hobbs’ definition, we describe an occasion relation as follows.

$$V_{ij} \text{ occ } V_{ij}$$

Each $V_{ij}$ is a discourse fragment that refers to value $V$ of feature $F_j$ of a common entity $E_i$, in different temporal situations $T_{ij}$, called initial, and $T_{ji}$, called final. This means that entity $E_i$ changes state between moments $T_{ij}$ and $T_{ji}$, resulting in the modification of the value of its feature $F_j$ as a consequence.

This has been captured as part of the proposed language as shown in Fig. 4.
As shown in Fig. 4, each occasion relationship involves two temporal situation references (initial and final), each one mapping a particular discourse fragment to one or more values. Although the class model in Fig. 4 cannot show it, the values mentioned by each of these discourse fragments must belong to a common entity. Thus, the change in the values over time (given by the different instances of Value), as well as the associated temporal sequence (given by the instances of TemporalSituationReference) can be expressed by the language.

B. Exemplification Relation

Following Hobbs’ definition, an exemplification relation is a coherence relation between pairs of discourse elements: “We let $S_I$ be the current clause and $S_0$ an immediately preceding segment. Thus, in an exemplification relation we can infer $p(A)$ from the assertion of $S_0$ and $p(a)$ from the assertion of $S_I$, where $a$ is a member or subset of $A$.”

Elaborating on Hobbs’ definition, we describe an exemplification relation as follows.

\[ V_j \text{ exe } W_{kl} \]

Each $V_j$ is a discourse fragment that refers to a value $V$ of a feature $F_j$ of an entity $E_i$. Each $W_{kl}$ is a discourse fragment that refers to a value $W$ of a feature $G_i$ of an entity $D_j$. Note that $G_i$ and $F_j$ can refer to the same features of the same entities, can match partially or differ completely. Similarly, $E_i$ and $D_j$ can refer to the same entities, can match partially or differ completely. $V_j$ are called “bases”, and $W_{kl}$ are called “examples”.

This means that one or several discourse fragments, playing the role of bases, can be exemplified by one or several discourse fragments, playing the role of examples. Note that the exemplification mechanism involves references to values of the same or different domain entities. Exemplification only occurs when the involved values belong to the same entity, or when they belong to different entities but these entities are strongly related. Since exemplifying means decreasing the level of abstraction (i.e. going from an abstract concept to a more concrete one), the relationships that make this possible must be those that implement an abstraction/concretion connection. As seen in [31], three major kinds of relationships must be considered: classification/instantiation, generalization/specialization and whole/part.

This has been captured as part of the proposed language as shown in Fig. 5.

As shown in Fig. 5, each exemplification relation involves two sets of references (the bases and the examples), each one mapping a particular discourse fragment to one value. Each value, in turn, belongs to a particular entity. As explained above, these entities may be related to each other through classification/instantiation, generalization/specialization or whole/part relationships, although the details of this are not captured in the language.

V. CASE STUDY: CIS REQUIREMENTS SPECIFICATION

In order to illustrate the proposed language and its benefits we present a case study in the requirements engineering field. Requirement engineering is an especially suitable area to apply discourse analysis due to its heavy influence in the final quality of the developed information system. In addition, requirements engineering is a well-known area with a large amount of practitioners, which makes it an ideal field where to use, validate, criticize and improve our proposal.

A significant number of authors have studied the relationships that occur between requirements [6, 32, 33]. An especially good example is [34], which identifies five categories of relations between requirements: structural, implementation, temporal, causality and necessity. However, very few of them link the requirements to the source text, unless they focus on information retrieval, in which case they usually avoid the formalization of their results as a language. In this section we show how the proposed language allows us to describe some requirements that emerge from a source text, while keeping them linked to it through the relevant semantic relations.

As explained in Section I, the final goal of the proposed language is to give structure to and extract semantic relations in the discourse analysis. In the specific application to requirement specifications, the proposed language presents two main benefits:
The proposed language allows the user to represent the documentation produced in the early stages of the requirements extraction process, which is typically written in a free-style text, as a highly-structured model. The structure of this model is extracted from the semantics of said documentation.

The proposed language allows the user to identify semantic relations presented in the text in an explicit and unambiguous way. These relations are crucial for the requirement engineer because they indicate aspects of the requirements that may be otherwise difficult to detect, such as causal dependencies between requirements, constraints or state changes.

We have chosen as a case study a fragment of a requirements specification document that is available online [35]. This specification describes the functionality required for an information system called CIS (Curricular Information System), intended to manage academic information for registration and course planning. The scope of the project, as well as the complete requirements specification mentioned above, is available online [36]. The discourse presented in the specification requirements has been analyzed and modeled using the proposed language. For this paper, we have selected the following fragment:

Currently, many requests between departments and Academic Services are done via email, phone calls, or interoffice mail. CIS allows many requests from Academic Services to the departments to be made via notices placed on the department administrator ‘portal’. For example, prior year final exam schedules are sent out to departments for them to review and update with requests for the current year’s final exam schedule. Instead a notice could be placed on the department ‘portal’: “It is time to schedule final exams....” And the information needed by the departments could be made available on-line for them to update and submit electronically.

Following Hobbs’ method, the above fragment is analyzed iteratively, looking for major breaks in the discourse and splitting the text into sentences with high semantic independence. In particular, the text has been divided into four sentences. In a second step, each sentence is analyzed, and its linguistic structure is matched to one of the available coherence relations. In our example, two coherence relations are applicable: the occasion relation and the exemplification relation: sentences involved in exemplification relations are very explicit, because they show example structures and clear textual references (such as “for example” phrases). Identifying instances of the occasion relation is more difficult to detect since it requires the identification of state changes and the associated entities and attributes that actually change value. The following paragraphs describe the analysis outcomes in full detail.

The final result of the analysis is shown in Fig. 6 as an object model, which includes the three areas that are present in the proposed language: the domain referred to by the discourse, the discourse elements themselves, and the coherence relations that map one another.

Regarding the discourse aspect, four objects (S1 to S4) have been created to represent each sentence in the text. S1 makes a reference of a very specific kind, namely a temporal situation reference (TSI), which describes the communication method between the departments and the Academic Services for that particular time frame. The necessary domain-related objects (V1, F1 and E1) have been created to support this. Most importantly, the TSI reference plays the “initial” role in an occasion relation, since the text it refers to describes an initial state of affairs for the involved entities.

S2, in turn, makes another temporal situation reference describing what the communication method between the departments and the Academic systems will be once CIS is implemented. This is implemented by linking TSI to V2, which establishes a new value for the same feature of the same entity, but at a later time. Finally, TSI plays the “final” role in the same occasion relation as TSI. In this manner, the occasion relation Occ1 is characterized through two temporal situation references (plus the associated domain elements), thus conveying the necessary change in state that is presented in the discourse.

S3 introduces an example about what is said in S1: the interaction for final exam scheduling is used to exemplify the current communication method between the departments and Academic Services. In order to capture this, we have created the entity E2 representing the specific interaction between departments and Academic Services for the final exam scheduling; this entity is a subtype of the more abstract E1, which represents any interaction between said parties. We have also created value V3 for feature F1 representing the fact that interaction for final exam scheduling is achieved by sending out emails or interoffice email.

The R1 reference, which maps the exemplifying text to the described domain entities, plays the role of “example” in the Exel exemplification relation. In turn, the “base” role is played by an R2 reference, which maps the previously mentioned S1 sentence to also existing domain elements. Note also that Exel illustrates the fact that a single discourse element (such as S1 in our case) can play a role in more than one coherence relation.

Finally, and working in a similar way as S3, S4 exemplifies a potential scenario that is supposed to happen in the temporal situation referred to in S2, i.e. once the new CIS is developed.

In summary, the case study here described reflects the intention of the writer of identifying a change of state in the communication method used between departments and Academic Services before and after the CIS is developed. Currently, the communication method is via email, phone calls or interoffice mail; with the new CIS, the communication method will change to portal notice. In addition, the writer wants to exemplify both situations (before and after). Final exam scheduling is used as an example of the current situation, and a possible scenario of CIS usage is used as an example of the future situation.
As we can see in the case study, the analysis of the discourse in the specification document provides structure to the information by using common, reusable constructs in the source text, domain and coherence relation areas. Also, a clear description of the different coherence relation and their mappings to text and domain is obtained. The model in Fig. 6, in fact, is a highly-structured depiction of the source text (discourse elements) with detailed mappings to the relevant software constructs (domain elements), plus an explanation of the rationale as why these mappings occur (coherence relations).

There are some additional benefits in the application of the proposed language to textual requirements specifications:

- Occasion relations may allow us to easily detect requirements that represent changes in the state of...
the domain. (as seen in the case study)

- The generalization and exemplification relations can point us to, respectively, general and specific cases of a particular requirement.
- The identification of causal relations or consequences between requirements may allow us to establish dependencies between them.
- Background relations may allow us to detect contextual constraints in the requirements which may be difficult to discover otherwise.

VI. OTHER APPROACHES

Apart from discourse analysis, there are existing approaches from the IS field to structure and extract semantic relations presented in non-structured information, focusing either on verbal communications or textual information.

Regarding verbal communications, communication analysis techniques [37] propose to undertake IS analysis from a communicational perspective [38]. These verbal approaches can be a complementary method when information is mixed verbal and textual, in order to improve the understanding of a specific domain. In addition, communication analysis can be used as a semantic validator of the models created such as that in Fig. 6.

In relation to textual information, the solutions are based on linked data [39] or metadata annotations. These approaches allow us to annotate pieces of discourse with semantic information, from simple annotations [40] to more complex ones [41]. These annotations try to accomplish similar goals to those of our work here. However, these approaches start from the premise that the text corpus being analyzed is also the source of the annotations being used, and therefore a high level of accuracy in the terminology is used [41]. For instance, the same semantic relation between elements can be annotate by different tags, such as “because”, “because of”, “so”, “cause” or “causal” in the case of a causal relation, and depending on what is found on the text itself. Although this approach results in highly adjusted models, its reusability and potential for abstract treatment is very low. The ISO/IEC 24744-derived language proposed here, on the contrary, is focused on creating an abstract language; following up with the above example, the application of this language allows us to realize that all the annotations such as “because” or “cause of” are all variations of a single kind of causal relation. This abstraction capability is very useful to identify coherence relations avoiding terminology differences.

These textual approaches could be improved by keeping their ability to tag text elements with very specific annotations, but adding and abstraction step that relates each of the used tags to a coherence relation as described in our approach.

VII. CONCLUSIONS

This paper introduces a language for discourse analysis, defined as an ISO/IEC 24744-derived metamodel. This language is designed to describe the semantic relations between elements in a text and the reality that it aims to describe. By using this language, information systems developers can explicitly describe the structure of the text and identify the underlying semantic relations between elements in the text and entities, values or types in the domain. In addition, the language can be easily integrated into a larger information systems development methodology that is based on the ISO/IEC 24744 standard.

In addition, the paper shows an application of the proposed language to the domain of requirements engineering, illustrating the benefits of the approach. This is done through the analysis of a text fragment and the construction of a highly structured, semantically-rich conceptual model that links the identified entities and values in the domain to elements in the source text. This scenario demonstrates how requirements engineering can be assisted through the linguistic analysis of text and the subsequent maintenance of the mappings between said text and the domain. For example, clauses describing temporal changes or causal lines of reasoning are easily mapped to specific domain configurations, so that semantics are formally preserved.

Furthermore, the proposed language adopts a domain-independent approach that covers the major coherence relations described in linguistic corpus. For this reason, we argue that it can be applied to the analysis of different kinds of documents and facilitate the integration of discourse analysis techniques in the information systems development process.

The application of the language provides a way to document a free-form text in a highly-structured manner, so that specific elements of the text are mapped to the corresponding software constructs through the relevant explanatory links (i.e. coherence relations). Since a model constructed with the language provides an explanation of why certain software artefacts exist, in terms of the discursive rationale that generates them, we argue that the proposed language is a Type II theory (i.e. explanatory theory) according to [42].

As future work, we intend to explore the incorporation of the necessary language constructs so that the relationships between entities in exemplification relations can be expressed from a more formal perspective, rather than being left out of the language. Also, we believe that the validation of the language through a wide range of case studies is an important challenge, since it will allow us to ascertain the capabilities and limitations of the language not only in requirements engineering but also in other areas. The identification, characterization and incorporation to the language of new coherence relations that may be relevant to information systems development is also an expected future line of work.

ACKNOWLEDGMENTS

This work has been partially funded by Spanish National Research Council through contract for the program “Junta de Ampliación de Estudios” (B.O.E. 146, 20/06/2011) and the FP7 INFRASTRUCTURES project “ARIADNE” (grant number 313193).

REFERENCES

