The ISO Standard for Dialogue Act Annotation, Second Edition

Harry Bunt[1], Volha Petukhova[2], Emer Gilmartin[3], Catherine Pelachaud[4], Alex Fang[5], Simon Keizer[6], and Laurent Prévot[7]
[1]Tilburg University, [2]Saarland University, Saarbrücken [3]Trinity College, Dublin, [4]CNRS-ISIR, Sorbonne University, Paris
[5]City University, Hong Kong [6]Toshiba Research Erope Ltd., Cambridge [7]Université Aix-Marseille
bunt@uvt.nl, v.v.petukhova@gmail.com, gilmare@tcd.ie, catherine.pelachaud@upmc.fr,
alex.fang@cityu.edu.hk, keizer.simon@gmail.com, laurent.prevot@lpl-aix.fr

Abstract

ISO standard 24617-2 for dialogue act annotation, established in 2012, has in the past few years been used both in corpus annotation and in the design and implementation of components of spoken and multimodal dialogue systems. This has brought some inaccuracies and undesirable limitations of the standard to light, which are addressed in the second edition. This second edition allows a more accurate annotation of dependence relations and rhetorical relations in dialogue. Moreover, a triple-layered plug-in mechanism is introduced which allows dialogue act descriptions to be enriched with information about their semantic content and other information, such as accompanying emotions, and which allows the annotation scheme to be customised by adding application-specific dialogue act types.

Keywords: dialogue annotation, dialogue acts, ISO standards, plug-ins, semantic annotation

1. Introduction

The first edition of ISO standard 24617-2 for dialogue act annotation, published in 2012 (Bunt et al 2010; 2012), has been applied in corpus annotation[1] and in the design of components for language understanding, dialogue management, and output generation in spoken and multimodal interactive systems.[2]. These applications have brought some inaccuracies of the standard to light and some undesirable limitations, both of which are addressed in the second edition.

The design of the standard has also contributed to the specification of a framework for defining other standards for semantic annotation. This framework was in turn established as an ISO standard in 2016 (ISO 24617-6, Principles of semantic annotation). In its specification the second edition of ISO 24617-2 closely follows the requirements and recommendations of this framework.

When revising an annotation standard, an important issue concerns the compatibility between annotations according to the original and the revised version. It is desirable that 'old' annotations do not require re-annotation or conversion, for being valid according to the revised version, except where the revision concerns the correction of errors in the original version.The revised standard should thus preferably be 'downward compatible' with the original version. The second edition of ISO 24617-2 is indeed downward compatible with its first edition, which was made possible by the fact that the ISO scheme is extensible in the following respects:

Dimensions: Due to the orthogonality of the set of dimensions, additional dimensions may be introduced as long as they are orthogonal to the already existing dimensions and to each other. (By the same token, dimensions may be left out without affecting the use of the remaining dimensions.)

Communicative functions: The taxonomy of communicative functions expresses semantic relations between functions: dominance relations express different degrees of specialisation; sister relations express mutually exclusivity. Communicative functions may be added to the taxonomy as long as these relations are respected.

Qualifiers: Of the attribute-value combinations that form qualifiers which can be associated with communicative functions, the attributes are 'orthogonal' in dealing with semantically distinct aspects, and the values are mutually exclusive. Additional attributes and values may be introduced as long as they respect these properties.

The second edition of ISO 24617-2 contains the following new elements:

- 'reference segments' for more accurate annotation of a dialogue act's dependence relations;
- the dimension of 'Contact Management', and communicative functions specific for this dimension;
- a mechanism for using 'layered' plug-ins with structured interfaces to ISO 24617-2 for enriching the description of individual dialogue acts and dialogue act sequences;
- predefined plug-ins and interfaces for:
  - rhetorical relations between dialogue acts;
  - the semantic content of dialogue acts;
  - fine-grained communicative functions for feedback;

[1]See e.g. Fang et al., 2012; Chowdhury et al. (2014); Petukhova et al. (2014), Gilmartin et al., 2018; Ngo et al. (2018); Bunt et al. (2019); Mezza et al. (2019).

[2]See e.g. Keizer and Rieser (2018); Keizer et al. (2019), Malchanau (2019), Malchanau et al. (2019)

- application-specific communicative functions;
- emotional aspects of dialogue acts.

ISO 24617-2 has remained fixed since its publication in 2012, as is appropriate for a good standard. The DIT++ annotation scheme, on which the ISO scheme is based, has evolved in order to accommodate inaccuracies and limitations encountered in use, The changes in ISO 24617-2 have first been implemented in DIT++ Release 5.2 (see `https://dit.uvt.nl/#Release5.2`).

This paper describes the changes in the ISO standard. Section 2 documents the changes relating to the accurate markup of semantic dependence relations in dialogue. Section 3 discusses the introduction of new dimensions and communicative functions. Section 4 describes the different use cases of the ISO and DIT++ annotation schemes and the requirements that each use case brings. Section 5 discusses the limitations of annotation standards, analysing the motivations for looking for ways to extend their coverage. The most important innovation is the use of structured 'plug-ins' with an abstract syntax and a semantics, according to the ISO 24617-6 principles of semantic annotation. Section 6 describes the formal structure of a layered plug-in and its interface with a host annotation scheme, and a number of predefined plug-ins for ISO 24617-2. Section 7 summarises the paper and concludes with perspectives for future work.

2. Dependence Relations in Dialogue

2.1. Types of Dependence Relation

ISO 24617-2 distinguishes two types of semantic dependence relations in dialogue: functional dependence relations and feedback dependence relations.
The functional dependence relation is defined as the *"relation between a given dialogue act and a preceding dialogue act on which the semantic content of the given dialogue act depends due to its communicative function."* This relation occurs with inherently responsive dialogue acts such as answers, (dis-)confirmations, (dis-)agreements, corrections and the acceptance or rejection of requests, offers, and suggestions. Such dialogue acts depend for their full meaning on one or more dialogue acts that occurred earlier in the dialogue. The property of 'responsiveness' is closely related to what is sometimes called 'backward-looking'; for example, in the DAMSL annotation scheme the communicative functions are divided into forward-looking and backward-looking. Backward-looking functions are defined as functions that indicate how the current utterance relates to the previous discourse. These also include feedback acts and acts concerned with speech editing.

The feedback dependence relation is defined as the *"relation between a feedback act and the stretch of communicative behaviour whose processing the act provides or elicits information about"*. Feedback acts provide information about the processing of what was said before - such as its perception or its interpretation. This is illustrated by the examples in (1).

(1) 1. A: Judging by the CVs, John Shlakeyin seems
 the best candidate to me.
 2. a. B: John WHO?
 b. B: I see.
 b. B: Agreed.

A feedback dependence relation targets one or more preceding *dialogue acts* if the feedback concerns high-level processing, such as understanding and agreeing (as in 2b and 2c), and it targets a dialogue *segment* in the case of low-level processing, such as hearing what was said (as in 2a). In the latter case, ISO 24617-2 stipulates that the feedback dependence relation should refer to the smallest functional segment containing the segment that the feedback act is about. This way of annotating feedback dependence relations is not quite accurate, since feedback about a stretch of communicative behaviour smaller than a functional segment is not about the entire segment. For example, negative feedback that signals a problem in hearing certain words may imply positive feedback about the rest of the segment. For more accurate annotation, the second edition introduces a *'reference segment'* as a stretch of communicative behaviour that is the target of a feedback dependence relation and that is not a functional segment.
Feedback acts have either a communicative function that is specific for feedback (i.e. *AutoPositive, AutoNegative, AlloPositive, AlloNegative, FeedbackElicitation*) or a general purpose communicative function, such as *CheckQuestion*, or *Confirm*. The two varieties are illustrated in (2). A feedback act with a responsive general-purpose function has a functional dependence; all other feedback acts have a feedback dependence.

(2) 1. C: Best before nine on Monday, or on Tuesday
 2. S: Monday before nine you said?
 [Auto-Feedback, CheckQuestion, feedback dependence on reference segment "Best before nine" in 1]
 3. C: That's right.
 [Allo-Feedback, Confirm, functional dependence on dialogue act expressed by 2]

2.2. Self- and Partner Repair

Reference segments are also useful for the accurate annotation of Own Communication Management (OCM) acts and Partner Communication Management (PCM) acts. For example, the accurate annotation of a self-correction (in the OCM dimension) or a partner correction (in the PCM dimension) requires the specification of the segment that is corrected, which may well be a single word or morpheme (rather than a functional segment).

3. Adding Dimensions

As mentioned in the Introduction, thanks to the independence ('orthogonality') of the dimensions, new dimensions may freely be added or left out without affecting the rest of the annotation scheme. The Contact Management dimension, known from DIT++, and known to be orthogonal to the other 9 dimensions (*Task, Turn Management, Time Management, Auto-* and *Allo-Feedback, Own* and *Partner*

Communication Management, Discourse Structuring, and *Social Obligations Management*), has been found to be needed when applying the ISO scheme and is added in the second edition, along with a few communicative functions specific for this dimension.

Task Management, another potential dimension known from DAMSL, has also been considered for inclusion in the ISO scheme. In DAMSL, this dimension was introduced for talking about the task in task-oriented dialogues. This includes utterances that involve coordinating the activities of the speakers (*Are you keeping track of the time?*), or discussing the status of the task (*Shall we start?* or *I think we're done*). An application where a great number of task management acts was found, is the DBOX corpus (Petukhova et al., 2014), where the task consists of participating in a quiz with the system in the role of quiz master; these dialogues have a first part in which the quiz master explains the rules of the game, and a second part in which the game is played. The two parts of these dialogues seem to be concerned with different tasks, rather than with different dimensions, but dialogue acts concerned with task management sometimes occur in the middle of the dialogue (*"Is this question too general perhaps?"*) and are more conveniently categorised as belonging to a different dimension.
Dialogues in the medical domain also frequently contain utterances where health care professionals make their role explicit to a patient, and feel like they belong to another dimension than the dialogue acts that concern the medical issue under discussion. To deal with this phenomenon, either Task Management can be added as a dimension to the ISO scheme or can be made available by means of a plug-in.
Doctor-patient dialogues also tend to contain dialogue acts that serve to reassure or encourage a patient, and expressions of worry and concern. Such dialogue acts could be considered as forming a separate dimension, concerned with building and maintaining interpersonal relations. This is important not only in the medical domain, but also for example in educational settings (tutor-student, master-apprentice). For dealing with this kind of dialogue acts there is again a choice between (a) considering these dialogue acts as application-specific, and adding them to the ISO scheme through a plug-in, or (b) adding a dimension of 'interpersonal relation management'. No decision has yet been made on this subject (see also Section 6.3).

4. Use Cases

Annotation schemes may have other use cases than corpus material markup. The W3C recommendation EmotionML mentions three types of use cases (Baggia et al., 2014; Burkhardt et al., 2017):

(3)
1. Manual annotation of material involving emotionality, such as annotation of videos, speech recordings, faces, etc.;
2. Automatic recognition of emotions from sensors, including physiological sensors, speech recordings, facial expressions, etc., as well as from multimodal combinations of sensors;
3. Generation of emotion-related system responses, which may involve reasoning about the emotional implications of events, emotional prosody in synthetic speech, facial expressions and gestures of embodied conversational agents, etc.

Similarly, a dialogue act annotation scheme may be designed primarily for dialogue annotation, but may also be useful for the on-line recognition and generation of dialogue acts in interactive systems, like spoken dialogue systems. The different use cases of a dialogue act annotation scheme bring the following different requirements and desiderata.

UC1: Manual annotation has the advantage of producing annotations of the highest quality if performed by experts, but has the drawback of being costly and only feasible for limited amounts of data. Expert annotation delivers the highest quality of annotations since expert annotators are not only skilled in recognising the relevant features of communicative behaviour, but also have a wealth of context information, general world knowledge, and commonsense reasoning abilities to infer speaker beliefs and intentions. This enables expert annotators to assign fine-grained characterisations to segments of dialogue behaviour with high accuracy. To support manual annotation, the annotation scheme should therefore include fine-grained concepts with the level of detail that expert annotators can use. On the other hand, manual annotation is often done by annotators who do not have quite the skills of experts, partly because of the high cost, but also because it is interesting to know whether an annotation scheme can successfully be used by less skilled or even 'naive' annotators (Geertzen et al., 2008); for such annotators it is useful if the annotation scheme includes less fine-grained concepts. In both DIT++ and ISO 24617-2 these two requirements are met by using a hierarchically structured set of more and less specific communicative functions.

UC2: Automatic annotation of human-human dialogue, or of the user's contributions in a human-computer dialogue, typically lacks the general world knowledge and the skills of expert human annotators, and has limited access to context information - often only as far as represented in the dialogue history. Automatic annotation therefore in general cannot reliably characterise dialogue behaviour with the same level of detail as expert manual annotation. To effectively support automatic annotation, the annotation scheme should therefore contain concepts that are more coarse-grained than those needed for expert annotation. For example, human annotators are sometimes able to recognise the level of processing relating to feedback behaviour, distinguishing positive (multimodal) feedback expressing understanding from feedback expressing agreement by the way of nodding in combination with certain vocal sounds (Petukhova and Bunt, 2009). As in the previous use case, the hierarchical structure of the inventory of communicative functions, with fine- and coarse-grained functions, is helpful to make the annotation scheme suitable for both manual and automatic annotation.

UC3: Automatic recognition of dialogue acts in user behaviour in an interactive system is a similar task as automatic dialogue act annotation, except that in an interactive system the semantic contents of dialogue acts play a prominent role, often determined by structural properties of the application domain. For a given application, it may be beneficial to have a tight coupling between communicative functions and semantic content, and to define application-specific dialogue act types for specific types of content. For effectively supporting this use case, it may be beneficial to extend the annotation scheme with application-specific concepts. Plug-ins promise to be a useful mechanism for this purpose; see Section 6.3.2.

UC4: Generation of dialogue acts in an interactive system concerns the decision how to continue a dialogue when it is the system's turn, and this is the main task of the system's dialogue manager. This decision is typically a two-stage process, where the first stage is to decide on the communicative functions and semantic contents of one or more suitable dialogue acts, and the second is to decide on an appropriate realisation in linguistic, nonverbal, or multimodal form. In contrast with human dialogue participants, who may be somewhat vague or unspecific about their beliefs and intentions, a system's dialogue manager typically works with precise beliefs and goals, and generates, in the first of these two stages, dialogue acts with fine-grained communicative functions. This happens in particular for feedback acts, since the system may report a processing problem that it has encountered in great detail. The DIT++ taxonomy (Bunt, 2009) was originally developed as on the one hand an instrument in the analysis of dialogue structure and dialogue mechanisms, and on the other hand a basis for the design of dialogue management modules in interactive systems, and therefore includes very fine-grained feedback functions.

ISO 24617-2 was originally designed for supporting interoperable dialogue act annotation (use cases UC1 and UC2). Use cases UC3 and UC4 have been found to be potentially of equally great interest, however; with some extensions, the scheme has been applied in the implementation of the dialogue manager in the Madrigal system (see Keizer and Rieser, 2018; Keizer et al., 2019) and in the Virtual Negotiation Coach system (Malchanau et al., 2019; see also Malchanau, 2019). These cases call for the combination of communicative functions with elements from outside the scheme. For example, the generation of a response by a dialogue system involves the combination of a communicative function and a propositional content, and the generation of a certain emotion requires the specification of a dialogue act that carries the emotion. Plug-ins are proposed for these purposes; see Section 6.

5. Annotation Standards

5.1. Inherent Limitations

All annotation standards have certain inherent limitations, which can be grouped into three categories:

Scope: The scope of an annotation standards, defined by the class of phenomena they are meant to cover, sometimes presents undesirable limitations due to the fact that classes of linguistic or communicative phenomena are hard to define in such a way that there is no overlap or interference with phenomena that fall outside the scope. The annotation of information about time and events using TimeML, for example, runs into problems in the annotation of quantification (as in *"every Monday"* or *"most Wednesdays"*), and overlaps with semantic role labelling schemes for roles of a temporal character, like the start- and end times and the duration of an event. Likewise, PDTB annotation of rhetorical relations in text (Prasad et al., 2008) struggles with occurrences of negation. Such limitations restrict the applicability of annotation schemes in use case UC1.

Generality: Annotation standards are designed to be applicable across domains and applications, and therefore do not include concepts that are specific for certain domains or applications. This is part of their strength, but is sometimes also a weakness, especially for applications where it is deemed essential to include the use of specialised domain-specific concepts. Such a situation brings the need to customise the standard by adding the relevant specialised concepts.

Lack of consensus: Annotation schemes sometimes suffer from a lack of consensus among researchers about the way a certain class of linguistic phenomena should be described. Two prominent examples are the annotation of discourse relations (a.k.a. 'rhetorical relations') and of emotions.

Regarding the annotation of emotions, Baggia et al. (2014) note that *"Any attempt to standardise the description of emotions using a finite set of fixed descriptors is doomed to failure: even scientists cannot agree on the number of relevant emotions, or on the names that should be given to them. (...) Given this lack of agreement on descriptors in the field, the only practical way of defining an EmotionML is the definition of possible structural elements and their valid child elements and attributes, but to allow users to 'plug in' vocabularies that they consider appropriate for their work."*

Plug-ins like those for EmotionML are simply lists of terms. Such plug-ins are not directly useful for semantic annotation, since semantic annotations are meant to capture certain aspects of the meaning of the primary data. If the annotations themselves do not have a well-specified meaning, then the annotations can hardly be said to capture any meaning. For semantic annotation, the terms are not so important, but rather the *concepts* denoted by the terms. The ISO principles of semantic annotation (ISO 24617-6) therefore include the requirement of 'semantic adequacy', which stipulates that semantic annotations must have a well-defined semantics (Bunt & Romary, 2002).

5.2. Schema Architectures

The requirement that semantic annotations have a well-defined semantics, combined with the fundamental distinction between linguistic annotations and annotation representation formats (Ide and Romary, 2004), means that the definition of a semantic markup language should have three parts, specifying (1) the class of well-defined annotation structures; (2) a format for representing such structures; (3) their semantics. The first part is called an *'abstract syntax'*, the second is a so-called *'concrete syntax'*, and the third is a specification of the semantics of the structures defined by the abstract syntax.

In the specification of an abstract syntax, two types of structure are distinguished: *entity structures* and *link structures*. An entity structure contains semantic information about a segment of primary data and is formally a pair ⟨m, s⟩ consisting of a markable, which refers to a segment of primary data, and certain semantic information. A link structure contains information about the way two or more segments of primary data are semantically related. In the abstract syntax of the Dialogue Act Markup Language (DiAML) of the ISO 24617-2 scheme, an entity structure is a formal specification of a dialogue act, while link structures describe rhetorical relations between dialogue acts.

The annotation structures defined by the abstract syntax can be represented (or 'encoded') in a many ways; XML is the most popular representation format, but other formats, such as attribute-value matrices or annotation graphs (Ide and Bunt, 2010) are equally possible. Bunt et al. (2019) describe alternative representation formats for DiAML annotation structures and how they are formally related. Every concrete syntax must satisfy two requirements: (1) *completeness*, which means that every structure defined by the abstract syntax has a representation, and (2) *unambiguity*, which means that every representation structure encodes exactly one annotation structure. These requirements are formalized in the DiAML specification by the stipulation that the concrete syntax definition includes (1) an encoding function for the structures generated by the abstract syntax, and (2) a decoding function that works in the opposite direction.

Formally, the definition of a semantic annotation language is a triple $\langle AS, CS, Sm\rangle$, where AS is an abstract syntax, CS is a concrete syntax, and Sm is a semantics. Each of these components or 'layers' is further structured: the abstract syntax specification consists of a conceptual inventory (*CI*) and a class of well-formed annotation structures (*AC*), like pairs and triples of concepts from *CI* as well as nested pairs and triples; the concrete syntax is formed by a vocabulary (*VC*), a class of 'representation structures' (*CC*), and an encoding function ($_eF$) that defines a representation for each annotation structure with its inverse ('decoding' $_eF^{-1}$); the semantics consists of a model structure (*M*) and an interpretation function (*I*) of annotation structures. The formal definition of a semantic annotation scheme A_a is thus:

(4) $A_a = \langle AS_a, CS_a, Sm_a\rangle$
$= \langle\langle CI_a, AC_a\rangle, \langle VC_a, CC_a, {}_eF_a, {}_eF_a^{-1}\rangle, \langle M_a, I_a\rangle\rangle$

6. Layered Plug-ins

6.1. Plug-in Architecture

A plug-in for a given annotation scheme A_a is a way of extending the scheme. Given that semantic annotation schemes have the structure shown in (4), extensions have an impact on all three layers of the scheme. Annotators only use the annotation representations defined by the concrete syntax and need not be concerned with the levels of abstract syntax and semantics. However, an implemented annotation scheme comes with all the other components. In particular the DiAML specification and implementation comes with a decoding function that can be activated to compute the annotation structure encoded by an XML representation, and with an interpretation function that computes the meaning of the annotation structure (and thus, indirectly, of the XML representation). This is illustrated at the DialogBank website (https://dialogbank.uvt.nl/annotated-dialogues/), which contains ISO 24617-2 annotated dialogues, and where Python scripts are available for decoding DiAML-XML annotations, constructing the encoded annotation structures of the DiAML abstract syntax, en for encoding these annotation structures in other, tabular representation formats (see Bunt et al., 2019). A plug-in for such an annotation scheme therefore specifies extensions at all three levels of the host annotation scheme. The upshot of this is that a plug-in PL_p has the same triple-layered structure

(5) $PL_p = \langle AS_p, CS_p, Sm_p\rangle$
$= \langle\langle CI_p, AC_p\rangle, \langle VC_p, CC_p, {}_eF_p, {}_eF_p^{-1}\rangle, \langle M_a, I_a\rangle\rangle$

A layered plug-in is formally just another annotation scheme; it can be combined with a host annotation scheme through the unions of their components, like the union of the two conceptual inventories, the union of the two vocabularies, and so on. In many cases, the structures generated by host scheme and plug-in need to be integrated more tightly than by the union of their components, however, and this requires the specification of a *plug-in interface*. This is again a triple-layered specification which introduces link structures by which host and plug-in annotations are combined; see (6) below for the interface of a plug-in PL_c for specifying the semantic content of dialogue acts for DiAML as a host annotation scheme.

Layered plug-ins can be used in two ways: (a) predefined standard plug-ins can be referenced in host annotations, having the effect of keeping annotations conformant to the standard, as is done in EmotionML (see (11) in Section 6.4), and (b) user-defined plug-ins can serve specific applications, leading to annotations that are not in all respects conformant to the standard but providing optimal support in the use cases UC3 and UC4. In the next section some predefined plug-ins are briefly described; more details can be found on the DIT++ Release 5.2 website dit.uvt.nl.

6.2. Predefined plug-ins

6.2.1. Semantic Content

For the use cases UC3 and UC4 it is important to have information about the semantic content of dialogue acts. The degree of detail in which semantic content should be represented depends on the application domain. For some domains a simple representation as a list of attribute-value pairs may be adequate; for others a representation in terms of events with their participants, time and place may be more appropriate; for more advanced applications it may be necessary to take general aspects of natural language utterance meaning into account, including predicate-argument structures, quantifications and modifications.

The interface of a semantic content plug-in *PL*$_c$ for a host annotation scheme A_a can be defined as shown in (6). Its abstract syntax ${}_aAS_c$ introduces the *content link structure* as a pair $\langle a, c\rangle$ consisting of a dialogue act entity structure ('a') and a content entity structure ('c'); the concrete syntax specifies its XML encoding using a <contentLink> element, and the semantics specifies its meaning as the application of the interpretation $I_a(a)$ of the dialogue act structure 'a', defined by the semantics of the host annotation scheme, to the argument $I_c(c)$, defined by the plug-in semantics. This semantics reflects the dialogue act theory underlying the ISO annotation scheme, according to which the semantics of a full-blown dialogue act is an update operation on information states, defined (Bunt, 2014) by applying the semantics of the communicative function to the semantic content (computed as the interpretation of the content annotation).

(6) ${}_aY_c = \langle {}_aAS_c, {}_aCS_c, {}_aSm_c\rangle$, with:
${}_aAS_c = \langle {}_aCI_c, {}_aAC_c\rangle = \langle\emptyset, \{\langle a, c\rangle\}\rangle\rangle$
${}_aCS_c = \langle {}_aV_c, {}_aCC_c, {}_aF_c\rangle =$
$\langle\langle\emptyset, \langle\{$<contentLink> element$\}$,
${}_aF_c\rangle(\langle a, c\rangle)$ = <contentLink dialAct=${}_aF_c(a)$
semContent = ${}_aF_c(c)$/>$\rangle$
${}_aSm_c$=$\langle {}_aM_c, {}_aI_c\rangle$=$\langle\emptyset, {}_aI_c(\langle a, c\rangle) = I_a(a)(I_c(c))\rangle$

This plug-in interface is combined with the content plug-in and the host annotation scheme by the union of their components to form the extended annotation scheme $A_a + PL_c +_a Y_c$

Attribute-Value Plug-in

A simple domain-specific plug-in for semantic content described in terms of attribute-value pairs leads to content annotation in the style (7).

(7) P1: I'd like to leave around ten in the morning
(= markable m1)
<avContent xml:id="c1" target="#m1"
attribute="departureTime" value="10:00"/>

Underlying this representation is a conceptual inventory that lists the attributes and their possible values, and the definition of entity structures consisting of attribute-value pairs $\langle A_i, v_{ij}\rangle$. The semantics of such an entity structure can be defined as the property $\lambda x.A'_i(x) = v'_{ij}$, where $A'_i = I_{av}(A_i)$ and $v'_{ij} = I_{av}(v_{ij})$. Attribute-value pairs, their XML encoding, and their semantics define a very simple content plug-in PL_{av}. The content annotations that they allow can be linked using the interface (6) to dialogue act annotation representations as shown in (8).

(8) <dialogueAct xml:id="da1"target="#m1"
speaker="#s" addressee="#a" dimension="task"
communicativeFunction="inform"/>
<avContent xml:id="c1" target="#m1"
attribute="departureTime" value="10:00"/>
<contentLink dialAct="#da1" content="#c1"/>

Events and semantic roles

A more general content plug-in may be based on ISO standard 24617-4 ('SemAF-SR') for the annotation of semantic roles. The annotation scheme of this standard marks up semantic information related to the question "Who did what to whom?" by assigning semantic roles to the participants in an event.

SemAF-SR interprets annotations as expressing the occurrence (or denied occurrence, in case of a clause with negative polarity) of certain events with certain participants in certain roles. The plug-in consists in this case of the abstract and concrete syntax and the semantics of the SemAF-SR markup language. See `https://dit.uvt.nl` for details.

Quantifiers and Modifiers

A plug-in for semantic content is more general and more powerful as it takes more aspects into account of the meanings of phrases, clauses, sentences, and other natural language structures. On top of the identification of events with their time, place, and participants with their respective roles, the interpretation of quantifier and modifier structures forms the most important source of semantic information. The ISO standard under development (ISO-DIS 24617-12 (see Bunt, Pustejovsky & Lee, 2018; Bunt, 2020) can be the basis of a plug-in for this type of information.

6.2.2. Rhetorical Relations

Reflecting the lack of theoretical consensus in this area, ISO 24617-2 does not specify any particular set of relations to be used when marking up rhetorical relations between dialogue acts, Users of the ISO scheme have often employed a variant of the relation set defined in ISO standard 24617-8 ('DR-Core'). This is a set of 18 'core' relations that occur in many annotation schemes.

Using ISO 24617-2 (first edition), two problems were noted when annotating rhetorical relations. First, many rhetorical relations have two arguments that play different roles, for example, a Cause relation has a "Reason" and a "Result" argument. DiAML, the markup language of ISO 24617-2 and (DIT++), has no provision for indicating the roles in a rhetorical relation. The DR-Core scheme does have constructs as well as attributes and values for this purpose, which have been imported in the second edition and allow representation structures as shown in (10).

Second, rhetorical relations may occur in dialogue between two dialogue acts, or between their semantic contents, or between one dialogue act and the semantic content of another. This phenomenon is known in the literature as the ‘semantic-pragmatic’ distinction. Example (9) illustrates this.

(9) a. Ray is sick. He was attacked by a virus. [*sem.*]
b. Ray is sick. He beats his wife. [*pragm.*]

To enable making this distinction, the second edition of ISO 24617-2 introduces a ‘domain’ associated with rhetorical relations as illustrated in (10).[3]

(10) <drLink arg1=“#da2” arg2=“#da3” rel=“cause”
    relDomain=“pragmatic”>
      <argRole arg=“#da2” role=“result”/>
      <argRole arg=“#da3” role=“reason”/>
    </drLink>

Specifying a set of rhetorical relations is left in the second edition to a plug-in, which is very simple in this case since no new structures need to be introduced in the abstract or concrete syntax, and therefore no plug-in interface is required, just the specification of a set of relations with their argument roles and their semantics. Taking the DR-core as point of departure, a plug-in is predefined containing 19 relations, which are listed in Table 1 in the Appendix.

6.3. Additional communicative functions

6.3.1. Overview

A plug-in for adding certain communicative functions has a very simple specification, since no new entity structures or link structures are needed, but only the following components: (1) Abstract syntax: conceptual inventory that lists the new functions; (2) Concrete syntax: specification of XML function names, and encoding function assigning names to conceptual inventory items; (3) Semantics: the (DiAML-) context-update semantics for the new communicative functions.

6.3.2. Application-specific functions

Designed to be domain-independent, ISO 24617-2 does not include communicative functions that are specific for a certain application domain. All its communicative functions are either general-purpose functions or are specific for one of the dialogue control dimensions. While they form a powerful battery of functions for use in any application, some applications may benefit from the availability of additional, domain-specific communicative functions. This is another area where plug-ins can be useful. For example, in a negotiation domain one finds bids, counter-bids, accepts and rejects of bids, and so on. Such acts can be viewed as special cases of the general-purpose functions Offer and AddressOffer, and they would thus fit well within the taxonomy of the standard.

6.3.3. Functions for ‘Social’ Talk

Most annotation schemes have been designed for annotating dialogues that have a clear purpose, such as finding a train connection, designing a remote control, or finding a route on a map. Natural everyday conversations often do not have such a clear task as their motivation, but have a social purpose such as bonding or establishing a pleasant atmosphere.
Task-related dialogues often have an initial phase in which the participants exchange small talk before getting to the task, and such initial phases have often been omitted in dialogue corpora, where the initial small talk is viewed as occurring ‘before’ the ‘actual’ dialogue. An exception is the ADELE corpus of chat dialogues (Gilmartin et al., 2018). The dialogues in this corpus often have rather elaborate initial phases with greetings and discussions of each other’s health, and sometimes also an extended leavetaking phase with various kinds of greetings and well-wishing.
In order to annotate the dialogue acts in such phases in a satisfactory way, the DIT++ annotation scheme (Release 5.2) has been extended with some functions in the Social Obligations Management dimension, which form a plug-in for ISO 24617-2. Table 2 in the Appendix lists these functions and gives some examples.

6.3.4. Fine-grained feedback functions

For applications that instantiate use case UC4, i.e. for the generation of dialogue acts in an interactive system, it was noted to be useful to have more fine-grained feedback functions available than the rather coarse ones defined in ISO 24617-2:2012. More fine-grained functions are available in DIT++, where a distinction is made between five levels of processing at which feedback may be provided or elicited: (1) attention; (2) perception; (3) interpretation (understanding); (4) evaluation; and (5) execution. These functions define a simple plug-in in the same way as the ones for social talk. Table 3 in the Appendix lists the 10 fine-grained auto-feedback functions; fine-grained allo-feedback functions (10 feedback-providing and 10 feedback eliciting functions) have similar definitions, see `https://dit.uvt.nl`.

6.4. Adding Emotion

A dialogue act may be performed with an expression of an emotion, such as amusement, irritation, or disappointment. ISO 24617-2 has no provisions for annotating that.
The W3C recommendation EmotionML (Burkhardt et al., 2017; `https://www.w3.org/TR/emotionml/`) provides a flexible scheme, designed with the aim of being combined with other annotation schemes. It characterises emotions as complex entities, including ‘emotion categories’ such as “anger”, “happiness”, or “surprise”, an intensity value (called ‘valence’), and a confidence value, as well as various alternative other ways of describing emotions, notably in terms of ‘action tendencies’, ‘appraisals’, and multiple ‘dimensions’. An emotion annotation in EmotionML may have components of various categories to reflect the complexity of emotions; for instance, in the snippet (11), taken from the document `https://www.w3.org/TR/emotionml/`, an emotion is annotated as being a form of anger with elements of sadness and fear.

(11) <emotion category-set=“`http://www.w3.org/TR/emotion-voc/xml#big6`”>
    <category name=“sadness” value=“0.3”/>
    <category name=“anger” value=“0.8”/>

[3]This addition could also be useful for accommodating the two-dimensional approach to discourse relation definitions recently proposed by Crible and Degand (2019).

```
    <category name="fear" value="0.3"/>
</emotion>
```

As noted above, in view of the lack of consensus in the community, EmotionML gives users a choice to select a suitable emotion vocabulary plug-in for use in their annotations. In order to promote interoperability, EmotionML offers a number of alternative emotion vocabularies that can be used for this purpose, which are either commonly used in technological contexts or represent emotion models from the scientific literature. One of the best known repositories of the latter kind is Ekman's 'big six' (Ekman, 1972), a set of basic emotions that are recognised and produced in many cultures. Example (11) shows how this repository (or one of the others listed by EmotionML) is referenced in an annotation.

EmotionML is defined only at the level of concrete syntax, but it can be used as the basis for defining a plug-in for ISO 24617-2. An emotion has an experiencer and an object that the emotion is about. The emotional aspect associated with a dialogue act is a relation between the speaker, as the experiencer of the emotion, and (the semantic content of) the dialogue act as the object of the emotion. For example, in (E13) the experiencer of the emotion associated with the acceptance of the preceding offer is participant P2 and the object is the semantic content of this offer and its acceptance, viz. P2 having a cup of coffee.

(12) P1: Would you like to have a cup of coffee?
(= markable m1)
P2: That would be wonderful! (= markable m2)

```
<dialogueAct xml:id"da1" target="#m1" speaker=
    "#p1" addressee="#p2" dimension=
    "SOM" communicativeFunction="offer"/S
<contentLink dialAct="#da1" content="#e1"/>
<dialogueAct xml:id="da2" target="#m2" speaker=
    "#p2" addressee="#p1" dimension="SOM"
    communicativeFunction="acceptOffer"
    functionalDependence="#da1"/>
<event xml:id="e1" target="#m2" pred=
    "have-coffee"/>
<srLink event="#e1" participant="#p2"
    semRole="agent"/>
<contentLink dialAct="#da2" content='#e1"/>
<emotion xml:id="em1" target="#m2"
    category="happiness" value="0.8"/>
<emoLink holder="#p2" object="#e1"
    emotion="#em1"/>
```

The annotation of emotions is not the primary aim of ISO 24617-2, but a simple plug-in for adding some information about emotion related to dialogue acts, based on EmotionML, can be defined as follows.
The abstract syntax lists a set of emotion categories and intensity values (any floating point number in the interval $[0, 1]$; entity structures as pairs $\langle m, \langle$ c, v$\rangle\rangle$ specifying an emotion category and an intensity value. The concrete syntax lists XML names for the emotion categories in the conceptual inventory and specifies encodings of entity structures using <emotion> elements (as defined in EmotionML, but simplified). The semantics interprets pairs $\langle$c, v$\rangle$ as attribute-value pairs where c denotes a two-place function, applicable to the experiencer and the object of an emotion, so $I_e(\langle$c, v$\rangle)$ is defined as the two-place predicate $\lambda x.\lambda y.I_e(c)(x, y) = I_e(v)$.

For linking emotion specifications to dialogue act annotations, a plug-in interface is needed that defines the <emoLink> element used in (12) with its underlying abstract syntax and semantics. In the abstract syntax, an emotion link structure is a triple $\langle$p, s, e$\rangle$ formed by a dialogue participant 'p' who is the sender of a dialogue act, the semantic content 's' of this dialogue act, and an emotion 'e'. These components correspond in the concrete syntax to the values of the attributes @holder, @object, and @emotion in an <emoLink> element, as illustrated in 12.

7. Concluding Remarks

The mechanism of triple-layered plug-ins, with an abstract syntax and a semantics and with a plug-in interface, allows the coverage of an annotation scheme to be extended with out-of-scope types of information and with application-specific concepts. The limitations of ISO 24617-2 and DiAML that users have noticed, are overcome in the second edition of the standard partly by defining some extensions and partly by plug-ins, including predefined plug-ins for various ways of expressing the semantic content of a dialogue act, for the choice of rhetorical relations, and for emotions associated with the performance of a dialogue act. For annotators there is not much difference between the availability of extensions and the use of plug-ins, but the latter provides greater flexibility and possibilities for customisability.

Future work will include the addition of annotations according to the second edition of the standard to the DialogBank (`https://dialogbank.uvt.nl/`, testing the customisability of the annotation scheme to dialogues in the medical domain, and evaluating the use of content plug-ins in the design of dialogue management systems.

8. Bibliographical References

Baggia, P., C. Pelachaud, C. Peter, and E. Zovato (2014). Emotion Markup Language (EmotionML) 1.0, W3C Recommendation 22 May 2014. `http://www.w3.org/TR/2014/REC-emotionml-20140522/`.

Bunt, H. (2009a). The DIT++ taxonomy for functional dialogue markup. In D. Heylen, C. Pelachaud, R. Catizone, and D. Traum (Eds.), *Proceedings of AAMAS-EDAML Workshop "Towards a Standard Markup Language for Embodied Dialogue Acts, Budapest"*, pp. 36–36.

Bunt, H. (2012). The semantics of feedback. In *Proceedings of SeineDial, 16th Workshop on the Semantics and Pragmatics of Dialogue*, Paris, pp. 118–127.

Bunt, H. (2013). A context-change semantics for dialogue acts. In H. Bunt, J. Bos, and S. Pulman (Eds.), *Computing Meaning, vol. 4*, pp. 177–201. Dordrecht: Springer.

Bunt, H. (2019). Plug-ins for content annotation of dialogue acts. In *Proceedings 15th Joint ISO-ACL Workshop on Interoperable Semantic Annotation (ISA-15)*, Gothenburg, Sweden, pp. 34–45.

Bunt, H. (2020). The annotation of quantification: The current state of ISO 24617-12. In *Proceedings of the 16th Joint ISO-ACL Workshop on Interoperable Semantic Annotation (ISA-16)*, Marseille.

Bunt, H., V. Petukhova, A. Malchanau, A. Fang, and K. Wijnhoven (2019). The DialogBank: Dialogues with interoperable annotations. *Language Resources and Evaluation 53*, 213–249. Available online at DOI: 10.1007/s10579-018-9436-9.

Bunt, H., J. Pustejovsky, and K. Lee (2018). Towards an ISO Standard for the Annotation of Quantification. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan.

Bunt, H. and L. Romary (2004). Standardization in Multimodal Content Representation: Some methodological issues. In *Proceedings of 4th International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, pp. 2219–2222. Paris: ELRA.

Burkhardt, F., C. Pelachaud, B. Schuller, and E. Zovato (2017). EmotionML. In D. Dahl (Ed.), *Multimodal Interaction with W3C Standards*, pp. 65–80. Cham (Switzerland): Springer.

Chowdhury, S., E. Stepanov, and G. Riccardi (2014). Transfer of corpus-specific dialog act annotation to the iso standard: Is it worth it? In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC 2014)*, Paris. ELRA.

Crible, L. and L. Degand (2019). Domains and functions: A two-dimensional account of discourse markers. *Discours 24*. Available online at DOI: 10.400/discours.9997.

Ekman, P. (1972). Universals and cultural differences in facial expressions of emotion. In J. Cole (Ed.), *Nebraska Symposium on Motivation, Vol. 19*, pp. 207–282. University of Nebraska Press.

Fang, A., J. Cao, H. Bunt, and X. Liu (2012). The annotation of the Switchboard corpus with the new ISO standard for dialogue act analysis. In *Proceedings 8th Joint ISO-ACL SIGSEM Workshop on Interoperable Semantic Annotation (ISA-8)*, ILC-CNR, Pisa.

Geertzen, J., V. Petukhova, and H. Bunt (2008). Evaluating dialogue act tagging with naive and expert annotators. In *Proceedings of the 6th Inernational Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech. Paris: ELRA.

Gilmartin, E., C. Saam, B. Spillane, M. O'Reilly, K. Su, A. Calvo, L. Cerrato, K. Levacher, N. Campbell, and V. Wade (2018). The ADELE corpus of dyadic social text conversations: Dialogue act annotation with ISO 24617-2. In *Proceedings 11th International Conference on Language Resources and Evaluation (LREC 2018).*

Ide, N. and H. Bunt (2010). Anatomy of annotation schemes: Mapping to GrAF. In *Proceedings 4th Linguistic Annotation Workshop (LAW IV)*, Uppsala., pp. 247–255.

Ide, N. and L. Romary (2004). International Standard for a Linguistic Annotation Framework. *Natural Language Engineering 10*, 211–225.

ISO (2012a). *ISO 24617-1: 2012, Language Resource Management - Semantic Annotation Framework (SemAF) - Part 1: Time and events*. Geneva: International Organisation for Standardisation ISO.

ISO (2012b). *ISO 24617-2:2012, Language Resource Management - Semantic Annotation Framework (SemAF) - Part 2: Dialogue acts*. Geneva: International Organisation for Standardisation ISO.

ISO (2014). *ISO 24617-4: 2014, Language Resource Management - Semantic Annotation Framework (SemAF) - Part 4: Semantic roles*. Geneva: International Organisation for Standardisation ISO.

ISO (2015). *ISO 24617-6:2015, Language Resource Management - Semantic Annotation Framework (SemAF) - Part 6: Principles of semantic annotation*. Geneva: International Organisation for Standardisation ISO.

ISO (2016). *ISO 24617-8:2016, Language Resource Management - Semantic Annotation Framework (SemAF) - Part 8: Semantic relations in discourse, Core annotation scheme (DR-Core)*. Geneva: International Organisation for Standardisation ISO.

Keizer, S. and H. Bunt (2006). Multidimensional dialogue management. In *Proceedings 7th SIGdial Workshop on Discourse and Dialogue*, Sydney, Australia, pp. 37–45.

Keizer, S., H. Bunt, and V. Petukhova (2011). Multidimensional dialogue management. In A. van den Bosch and G. Bouma (Eds.), *Interactive Multimodal Question Answering*, pp. 57–86. Berlin: Springer.

Keizer, S., O. Dusek, X. Liu, and V. Rieser (2019). User evaluation of a multi-dimensional statistical dialogue system. In *Proceedings 20th SIG*.

Malchanau, A. (2019). *Cognitive Architecture for Multimodal Multidimensional Dialogue Management*. Saarbrücken: PhD Thesis, University of Saarland.

Malchanau, A., V. Petukhova, and H. Bunt (2019). Towards integration of cognitive models in dialogue management: Designing the virtual negotiation coach application. *Dialogue and Discourse 9(2)*, 35–79.

Mezza, S., A. Cervone, E. Stepanov, G. Tortoretto, and G. Riccardi (2018). ISO-standard domain-independent dialogue act tagging for conversational agents. In *Proceedings of he 27th International Conference on Computational Linguistics (COLING 2018)*, Santa Fé, New Mexico, USA, pp. 3539–3551. Association for Computational Linguistics.

Ngo, T., K. Pham, and H. Takeda (2018). A Vietnamese dialogue act corpus based on the ISO 24617-2 standard. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. Paris: ELRA.

Petukhova, V. and H. Bunt (2009). Grounding by nodding. In *Proceedings GESPIN, Conference on Gestures and Speech in Interaction*, Poznán.

Prasad, R., N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. Joshi, and B. Webber (2008). The Penn Discourse TreeBank 2.0. In *Proceedings 6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech. Paris: ELRA.

Appendix

Table 1: Rhetorical relations, based on ISO 24617-8.

Relation	*Definition*
Cause	The second argument provides a reason why the first argument occurs or holds true.
Condition	The first argument is an unrealized situation which, when realized, would lead to the situation that forms the second argument.
Negative Condition	The first argument is an unrealized situation which, when not realized, would lead to the situation that forms the second argument.
Purpose	The second argument is the goal or purpose of the situation that forms the first argument.
Manner	The second argument describes how the first argument comes about or occurs.
Concession	The second argument cancels or denies an expected causal relation between the first argument and the negation of the second.
Contrast	One or more differences between the two arguments are highlighted with respect to what each predicates as a whole or about some entities they mention.
Exception	The second argument indicates one or more circumstances in which the situation that forms the first argument does not hold.
Similarity	One or more similarities between the two arguments are highlighted with respect to what each predicates as a whole or about some entities they mention.
Substitution	The two arguments are alternatives, the situation of the second argument being the favored or chosen alternative.
Conjunction	The two arguments bear the same relation to some other situation evoked in the discourse. Their conjunction indicates that they both hold with respect to that situation.
Disjunction	The two arguments bear the same relation to some other situation evoked in the discourse. Their disjunction indicates that they are non-exclusive alternatives with respect to that situation.
Exemplification	The second argument is a situation that is an element of the set of situations described by the first argument. Arg1 describes a set of situations.
Elaboration	The two arguments are the same situation, but the second argument is specified in more detail.
Restatement	The two arguments are the same situation, but viewed from different perspectives.
Synchrony	The two arguments form two temporally overlapping situations.
Asynchrony	The first argument temporally precedes the second.
Expansion	The two arguments are distinct situations that involve some shared entities; the second argument expands a narrative of which the first argument forms part of a certain narrative and Arg1 is a part, or expanding on the setting relevant for interpreting Arg1.
Evaluation	The second argument provides an opinion on the social, esthetic, economic, or other qualities of the first argument.

Table 2: Additional communicative functions for Social Obligations Management (from DIT++ Release 5.2).

Function	*Definition (S = sender, A = Addressee*	*Examples*
Follow-on Greeting	S wants A to know that S has established the presence and the identity of A.	"Hi Anne."
Politeness Question	S wants to know the state of well-being of A, or of someone close to A.	"How do you do?" "How is your mother?"
Opening Politeness Statement	S wants A to know that S is pleased to meet A.	"Nice to meet you"
Closing Politeness Statement	S wants A to know that S is pleased to have met A. S intends to soon close the dialogue.	"It was nice talking to you."
Farewell Wish	S wishes A to be in a positive state of well-being, and intends to close the dialogue.	"Have a good time".

Table 3: Fine-grained communicative functions for auto-feedback (from DIT++).

	Communicative Function	*Definition (in brief, S = speaker)*
1.	Attention Positive Auto-Feedback	S has noticed that something was said/done.
2.	Perception Positive Auto-Feedback	S has registered (heard/seen/felt,..) what was said/done.
3.	Interpretation Positive Auto-Feedback	S has interpreted what was said/done.
4.	Evaluation Positive Auto-Feedback	S has evaluated what was said/done.
5.	Execution Positive Auto-Feedback	S has acted on what was said/done.
6.	Attention Negative Auto-Feedback	S has failed to notice that something was said/done.
7.	Perception Negative Auto-Feedback	S has not been able to register (hear/see/feel,..) what was said/done.
8.	Interpretation Negative Auto-Feedback	S has not been able to interpret what was said/done.
9.	Evaluation Negative Auto-Feedback	S has not been able to evaluate what was said/done.
10.	Execution Negative Auto-Feedback	S has not been able to act on what was said/done.