INTERNATIONAL STANDARD

ISO 24617-2

Second edition
2020-12

Language resource management — Semantic annotation framework (SemAF) —

Part 2: Dialogue acts

Gestion des ressources langagières — Cadre d'annotation sémantique (SemAF) —

Partie 2: Actes de dialogue

Reference number
ISO 24617-2:2020(E)

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 37, *Language and terminology*, Subcommittee SC 4, *Language resource management*.

This second edition cancels and replaces the first edition (ISO 24617-2:2012), which has been technically revised.

The main changes compared to the previous edition are as follows:

— in 6.2, ‘reference segments’ are introduced to allow more accurate annotations of feedback dependence relations;

— in 6.3, a more detailed way of annotating rhetorical relations between dialogue acts is made possible by importing concepts from ISO 24617-8:2016 (DR-core);

— in 7.2, the Contact Management dimension, known from the DIT++ annotation scheme, and the Task Management dimension, known from the DAMSL annotation scheme, have been added, along with a few communicative functions specific for contact management;

— in 7.5 and Annex D, a possibility is introduced for importing elements from the W3C recommendation EmotionML in order to add affective information to dialogue acts;

— in Clause 9 and Annex D, the mechanism of ‘triple-layered annotation scheme plug-in’ with ‘plug-in interface’ is introduced; this mechanism allows the dialogue act annotation to be customized, using application-specific concepts, and to be enriched with semantic content information.

A list of all parts in the ISO 24617 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

Since its publication in 2012, ISO 24617-2 has been used in a number of annotation efforts as well as in the development of language-based interactive systems. These experiences have brought to light

— that the standard allowed dialogue act annotations that are slightly inaccurate in some respects,

— that some applications would benefit from the availability of mechanisms for customizing the set of concepts defined in the standard, and

— that certain use cases require the representation of functional dialogue act information to be extended with semantic content information.

This second edition seeks to remedy the noted inaccuracies, and to provide mechanisms

a) for customizing the set of defined concepts, and

b) for extending the information types in dialogue act annotations.

The improved accuracy of this second edition concerns the annotation of semantic dependence relations of dialogue acts and their scopes, and of rhetorical relations between dialogue acts. The mechanisms for extending and customizing the standard for a specific application concern most notably the annotation of information about the (domain-specific) semantic content of dialogue acts, the introduction of application-specific dialogue act types, the addition of communicative functions for fine-grained specification of feedback, and the annotation of speaker emotions.

This second edition is downward compatible with the original ISO 24617-2:2012 in the sense that every annotation made with the original version is a valid annotation according to the second edition. Existing annotations do not need to be revised in order to be compliant with this second edition.

Language resource management — Semantic annotation framework (SemAF) —

Part 2: Dialogue acts

1 Scope

This document provides a set of empirically and theoretically well-motivated concepts for dialogue annotation, a formal language for expressing dialogue annotations (the Dialogue Act Markup Language, DiAML), and a method for segmenting a dialogue into semantic units. This allows the manual or automatic annotation of dialogue segments with information about the communicative actions which the participants perform by their contributions to the dialogue. The annotation scheme specified in this document supports multidimensional annotation of spoken, written, and multimodal dialogues involving two or more participants. Dialogue units are viewed as having multiple communicative functions in different dimensions. The markup language DiAML has an XML-based representation format and a formal semantics which makes it possible to perform inferences with DiAML representations. This document also specifies data categories for dimensions of dialogue analysis, for communicative functions, for dialogue act qualifiers, and for relations between dialogue acts. Additionally, it provides mechanisms for customizing these sets of concepts, extending them with application-specific or domain-specific concepts and descriptions of semantic content, or selecting relevant coherent subsets of them. These mechanisms make the dialogue act concepts specified in this document useful not only for annotation but also for the recognition and generation of dialogue acts in interactive systems.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
addressee
dialogue (3.5) *participant* (3.13) oriented to by the *sender* (3.20) in a manner to suggest that his/her *utterances* (3.25) are particularly intended for this participant, and that some response is therefore anticipated from this participant, more so than from the other participants

Note 1 to entry: This definition is a *de facto* standard in the linguistics literature.

[SOURCE: Reference [34], modified - ‘speaker' replaced by ‘sender', and use of ambiguous pronouns avoided.]

3.2
allo-feedback act
feedback act (3.8) where the *sender* (3.20) elicits information about the *addressee's* (3.1) processing of an *utterance* (3.25) that the sender contributed to the *dialogue* (3.5), or where the sender provides information about his perceived processing by the addressee of an utterance that the sender contributed to the dialogue

EXAMPLE 1. A: Now move up.

2. B: Slightly northeast you mean?

3. A: Slightly yeah

With utterance 3, A performs an allo-feedback act signalling that he/she thinks B understood utterance 1 correctly.

3.3
auto-feedback act
feedback act (3.8) where the *sender* (3.20) provides information about his/her own processing of an *utterance* (3.25) contributed to the *dialogue* (3.5) by another *participant* (3.13)

EXAMPLE B's utterance in the example dialogue fragment in 3.2 signals that he/she is uncertain whether he/she understood the previous utterance correctly.

3.4
communicative function
property of certain stretches of communicative behaviour, describing how the behaviour changes the *information state* (3.12) of an understander of the behaviour

3.5
dialogue
exchange of *utterances* (3.25) between two or more persons or artificial agents

3.6
dialogue act
communicative activity of a *dialogue* (3.5) *participant* (3.13), interpreted as having a certain *communicative function* (3.4) and *semantic content* (3.18)

Note 1 to entry: A dialogue act can additionally also have certain *functional dependence relations* (3.10), *rhetorical relations* (3.17) and *feedback dependence relations* (3.9) with other units in a dialogue.

3.7
dimension
class of *dialogue acts* (3.6) that are concerned with a particular aspect of communication, corresponding to a particular category of *semantic content* (3.18)

EXAMPLE (1) Dialogue acts advancing the task or activity that motivates the dialogue (the 'Task' dimension); (2) dialogue acts providing and eliciting feedback (the Auto- and Allo-Feedback dimensions); (3) dialogue acts for allocating the speaker role (the Turn Management dimension).

3.8
feedback act
dialogue act (3.6) that provides or elicits information about the *sender's* (3.20) or the *addressee's* (3.1) processing of something that was uttered in the *dialogue* (3.5)

Note 1 to entry: Two classes of feedback are distinguished: *allo-feedback acts* (3.2) and *auto-feedback acts* (3.3).

3.9
feedback dependence relation
relation between a *feedback act* (3.8) and the stretch of communicative behaviour the processing of which the act provides or elicits information about

EXAMPLE In the example in 3.2, both the allo-feedback act expressed by utterance 3 and the auto-feedback act expressed by utterance 2 have a feedback dependence relation to utterance 1.

Note 1 to entry: Feedback dependence relations are also used to relate self-corrections, partner corrections, and other speech editing acts, which strictly speaking are not feedback acts, to the segments that they apply to.

3.10
functional dependence relation
relation between a *dialogue act* (3.6) with a *responsive communicative function* (3.16) and one or more previous dialogue acts that it responds to

EXAMPLE The relation between an answer and the corresponding question, such as between utterance 3 and utterance 2 in the example in 3.2; or the relation between the acceptance of an offer and the corresponding offer.

3.11
functional segment
minimal stretch of communicative behaviour that has one or more *communicative functions* (3.4)

Note 1 to entry: The condition of being 'minimal' ensures that functional segments do not include material that does not contribute to the expression of a communicative function that identifies the segment.

EXAMPLE The functional segment corresponding to the answer given by S in the following dialogue fragment does not include the parts *"Just a moment please"* and *".... let me see..."* but only the parts *"the first train to the airport on Sunday morning is"* and *"at 5:45"*.

1. U: What time is the first train to the airport on Sunday morning please?

2. S: Just a moment please... the first train to the airport on Sunday morning is .... let me see... at 5:45.

Note 2 to entry: A consequence of this definition is that functional segments can be discontinuous, can overlap or be embedded, and can contain parts from more than one turn.

3.12
information state
context
totality of a *dialogue* (3.5) *participant's* (3.13) beliefs, assumptions, expectations, goals, preferences, hopes, and other attitudes that may influence the participant's interpretation and generation of communicative behaviour

3.13
participant
person or artificial agent involved in the exchange of *utterances* (3.25)

3.14
qualifier
predicate that can be associated with a *communicative function* (3.4)

EXAMPLE A: Would you like to have some coffee?

B: Only if you have it ready.

B's utterance accepts A's offer under a certain condition; this can be described by qualifying the communicative function Accept Offer with the predicate 'conditional'.

3.15
reference segment
stretch of communicative behaviour that a *feedback dependence relation* (3.9) refers to and that is not a *functional segment* (3.11)

3.16
responsive communicative function
communicative function (3.4) of a *dialogue act* (3.6) that depends for its *semantic content* (3.18) on one or more dialogue acts that it responds to

Note 1 to entry: See 5.2.

Note 2 to entry: In 7.3.4, the set of responsive communicative functions is listed of the annotation scheme defined in this document.

3.17
rhetorical relation
discourse relation
semantic or pragmatic relation between two *dialogue acts* (3.6) or their *semantic contents* (3.18)

Note 1 to entry: Relations such as *elaboration, explanation, justification, cause,* and *concession* have been studied extensively in the analysis of (monologue) text, where they are often called 'rhetorical relations' or 'discourse relations', and are mostly viewed either as relations between text segments or as relations between events or propositions, described in text segments. Many of these relations also occur in *dialogue* (3.5).

EXAMPLE 1 In the following example, the statement in the second utterance provides a *motivation* for the question in the first utterance:

A: Can you tell me what flights there are to Sydney on Saturday? I'd like to attend my mother's 80th birthday.

EXAMPLE 2 A rhetorical relation between the semantic contents of two dialogue act occurs in the following, where the content of B's statement mentions a *cause* for the content of A's statement:

A: I can never find these stupid remote controls.

B: That's because they don't have a fixed location.

3.18
semantic content
information, situation, action, event, or objects that a stretch of communicative behaviour refers to

3.19
semantic content category
semantic content type
type of the *semantic content* (3.18) of a *dialogue act* (3.6)

EXAMPLE The various *dimensions* (3.7) defined in this document correspond to categories of semantic content. In particular, the Task dimension corresponds to the category of task-specific actions and information; the Allo- and Auto-Feedback dimensions correspond to the categories of information about the processing by the current speaker or by the addressee, respectively, of something that was said before; the Turn Management dimension corresponds to the category of information about the allocation of the speaker role, and so forth.

3.20
sender
dialogue (3.5) *participant* (3.13) who performs a *dialogue act* (3.6)

3.21
speaker
sender (3.20) of a *dialogue act* (3.6) in spoken form, possibly combining speech with nonverbal communicative behaviour

Note 1 to entry: A *dialogue* (3.5) *participant* (3.13) can contribute to a dialogue without having the *speaker role* (3.22), for example by nodding in agreement to what the other participant says. Therefore, the term 'speaker' is not synonymous with 'participant who occupies speaker role'.

3.22
speaker role
role occupied by a *participant* (3.13) who has temporary control of a *dialogue* (3.5) and speaks for some period of time

[SOURCE: DAMSL annotation scheme (see Reference [3]).]

3.23
speech act
act that a *speaker* (3.21) performs when producing an *utterance* (3.25)

Note 1 to entry: The notion 'utterance' in this definition is commonly interpreted as mentioned in Note 1 to entry of 3.25.

[SOURCE: SIL Glossary of linguistic terms (https://glossary.sil.org/term/speech-act), modified - Added Note 1 to entry.]

3.24
turn unit
stretch of communicative activity produced by one *participant* (3.13) who occupies the *speaker role* (3.22), bounded by periods of inactivity of that *sender* (3.20) or by periods where another participant occupies the speaker role

Note 1 to entry: The term 'turn unit' corresponds to one of the meanings of the often used term 'turn', which is ambiguous between 'turn unit' and 'right to speak", as in "to have the turn" and "turn-taking". The term 'turn' is only used in this document when the context makes it clear in what sense the term is meant.

Note 2 to entry: The term 'turn unit' is also closely related to the term 'turn construction unit' (TCU), introduced by Reference [51]. The TCU seems a rather intuitive and holistic notion, of which the usefulness has been the subject of debate (see e.g. Reference [52]). The term is therefore avoided in this document.

Note 3 to entry: The term 'turn unit' is useful in the description of *dialogue* (3.5) behaviour, but is not of central importance in this document, since *dialogue acts* (3.6) are not assumed to correspond to turn units.

3.25
utterance
anything said, written, keyed, signed, or otherwise expressed, possibly in multimodal form

Note 1 to entry: An utterance is part of a *turn unit* (3.24). In the literature, the term is commonly used in the sense of '*everything* contributed by a *sender* (3.20) within a turn unit'.

Note 2 to entry: The term 'utterance' is useful in the description of *dialogue* (3.5) behaviour, but is not of central importance in this standard, since *dialogue acts* (3.6) are not assumed to correspond to utterances, but rather to the communicative behaviour in *functional segments* (3.11).

4 Use cases

The notion of a dialogue act plays a key role in the analysis of spoken and multimodal dialogue, as well as in the design of spoken dialogue systems and embodied conversational agents. These applications all depend on the availability of dialogue corpora, annotated with dialogue act information. The main purpose of this document is to define a reference set of domain-independent basic concepts for dialogue act annotation, and their use for representing such annotations, in the Dialogue Act Markup Language (DiAML). The set of concepts defined in ISO 24617-2:2012 was based on the DIT++ taxonomy, which was originally developed to serve a double purpose: the articulate functional description of communicative activity in natural human dialogue, and a basis for the design of modules in dialogue systems. As part of the ISO 24617 series, a focus came to lie on annotation. Still, like DIT++, this document has multiple use cases, which can be grouped into four types:

— UC1: manual annotation of spoken, written, or multimodal human-human or human-computer dialogue;

— UC2: automatic annotation of spoken, written, or multimodal human-human or human-computer dialogue starting from transcriptions or recordings of communicative behaviour;

— UC3: recognition of dialogue acts in (multimodal) communicative user behaviour;

— UC4: generation of dialogue acts by the dialogue manager component of a dialogue system.

The different use cases bring different requirements and desiderata, as follows.

— UC1: manual annotation is costly and only feasible for limited amounts of data, but has the advantage of producing annotations of the highest quality since expert annotators have a wealth of context information, general world knowledge, and common-sense reasoning abilities to infer speaker beliefs and intentions. Expert annotators are therefore able to make fine-grained annotations. In order to support this use case, the annotation scheme should include fine-grained concepts.

— UC2: automatic annotation systems typically cannot characterize dialogue behaviour with the same level of detail as expert human annotators, since they lack common world knowledge, and usually have access to context information only as far as represented in the dialogue history. Automatic annotation is therefore in general less fine-grained. To effectively support this use case, the annotation scheme should contain more coarse-grained concepts than those needed for use case UC1.

— UC3: recognition of dialogue acts by an interactive system is almost the same as automatic dialogue act annotation, except that in an interactive system the semantic contents of dialogue acts play a prominent role. For a given application, it may be beneficial to define application-specific functions for specific types of content. For effectively supporting this use case, it may be useful to extend the annotation scheme with application-specific concepts.

— UC4: generation of dialogue acts in an interactive system concerns the decision how to continue a dialogue, and this is the main task of a dialogue manager component. This task is typically organized as a two-stage process: (1) decide on the communicative functions and semantic contents of one or more possible dialogue acts; (2) decide on a realization in an appropriate 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 stage (1), dialogue acts with fine-grained communicative functions, for example for feedback acts, since the system may report a processing problem with great accuracy. This calls for the annotation scheme to include very fine-grained functions.

The new elements in this second edition of this document were introduced for providing more effective support for each of these use cases, in particular for the cases UC3 and UC4.

5 Basic concepts and metamodel

5.1 Dialogue acts

The term 'dialogue act' is often used rather loosely in the sense of speech act used in dialogue. Indeed, the idea of interpreting communicative behaviour in terms of actions, such as questions, promises, and requests goes back to speech act theory [9],[50]. But where speech act theory is primarily an action-based approach to meaning within the philosophy of language, dialogue act theory is an empirically-based approach to the computational modelling of linguistic and nonverbal communicative behaviour in dialogue.

Dialogue acts offer a way of characterizing the meaning of communicative behaviour in terms of update operations, to be applied to the information states of participants in the dialogue; this approach is commonly known as the 'information-state update' or 'context-change' approach; see e.g. References [12] and [56]. For instance, when an addressee understands the utterance *"Do you know what time it is?"* as a question about the time, then the addressee's information state is updated to contain (among other things) the information that the speaker does not know what time it is and would like to know that. If, by contrast, it is understood that the speaker is reproaching the addressee for being late, then the addressee's information state is updated to include (among other things) the information that the speaker *does* know what time it is. Distinctions such as that between a question and a reproach concern the *communicative function* of a dialogue act, which is one of its two main components. The other main component is its *semantic content*, which describes the objects, properties, relations, situations, actions or events that the dialogue act is about. The communicative function of a dialogue act specifies how an addressee updates his information state with the information expressed in the semantic content when he/she understands the dialogue act.

This approach to the definition of communicative functions is strictly semantic, in contrast to approaches based on linguistic form. For example, the behaviour of a speaker who repeats something that was said by someone else may be characterized as a 'repetition' (which is a communicative function in some annotation schemes); however, this only says something about the *form* of the behaviour compared to the repeated behaviour, not about its function. A repetition often has a feedback function, as in (1.2), but it can also have other functions, as in (1.4), where it is used as a confirmation in response to a check question.

(1) 1. S: There are evening flights at seven-fifteen and eight-thirty.

2. C: Seven-fifteen and eight-thirty.

3. C: And that's on Sunday too.

4. S: And that's on Sunday too.

A form-related requirement for introducing a communicative function is however that there are observable features of communicative (linguistic and/or nonverbal) behaviour which are indicative for that function in the context in which the behaviour occurs. This requirement puts all communicative functions on an empirical basis.

Dialogue act annotation is the marking up of stretches of dialogue with information about the dialogue acts they contain. Spoken dialogues are traditionally segmented into *turns*, in the sense of 'turn units' as defined in 3.24. Such turns can be quite long and complex, and are therefore not the most useful units of behaviour to assign communicative functions to. Communicative functions can be assigned more accurately to smaller units that are functionally relevant. Such units are called *functional segments*, and are defined as the minimal stretches of communicative behaviour that have one or more communicative functions. Subclause 6.3 discusses dialogue segmentation.

Inherent to the notion of a dialogue act is that there is an agent who produces the dialogue act, called the 'sender', and one or more agents who are addressed, called the 'addressee(s)'. Dialogue studies often focus on two-person dialogues, in which case the dialogue acts have only one addressee. Besides sender and addressee(s), there may be various types of side-participants who are present but do not or only marginally participate[29].

Dialogue act annotation is often limited to assigning communicative functions to dialogue segments, which corresponds intuitively to indicating the type of communicative action that is performed. A semantically more complete characterization additionally provides information about the category of semantic content. The DAMSL annotation scheme distinguishes three categories of semantic content: Task, Task Management, and Communication, which indicate whether the semantic content of the dialogue act advances the task which underlies the dialogue, or discusses how to perform the task, or concerns the communication process. The DIT++ scheme distinguishes a number of subcategories of communication-related information, such as feedback information, turn allocation information, and speech management information. The scheme in this document inherits the DIT++ categories of semantic content, also called 'dimensions'; see Clause 7.

Example (2) illustrates the use of the key attributes of a dialogue act in the DiAML-XML annotation of a task-related yes-no question addressed by speaker 'a' to addressee 'b', expressed by the functional segment 'm1'.

(2)
```
<dialogueAct xml:id="da1" target="#m1" sender="#a" addressee="#b" dimension="task"
      communicativeFunction="propositionalQuestion"/>
```

5.2 Dependence relations

Some types of dialogue acts are inherently dependent for their full meaning on one or more dialogue acts earlier in the dialogue, which they respond to. This is for example the case for answers, whose meaning is partly determined by the question that is being answered, and also for the acceptance or rejection of offers, suggestions, requests, and apologies. This is illustrated in example (3), where the meaning of the answer in turn 3 depends on whether it is an answer to the question in turn 1 or to the one in turn 2.

(3) 1. B: Do you know who's coming tonight?

2. B: Which of the project members do you think will be there?

3. A: I'm expecting Jan, Alex, Claudia, and David, and maybe Olga and Andrei.

As an answer to the question in 1, A's answer says that nobody else is expected to come than the people that are mentioned, but as an answer to the question in 2 it leaves open the possibility that other people will come, who are not members of 'the project'.

This kind of semantic dependence, which is due to the responsive character of some communicative functions, is called a *functional dependence relation*. Marking up this relation between a dialogue act with a responsive communicative function and its 'antecedent' dialogue acts allows the annotation to not just indicate e.g. that an utterance has the function of an answer, but also to indicate *to which question* it is an answer, as illustrated in (4). Subclause 7.3.4 lists the responsive communicative functions defined in this document.

(4) a. B: Which of the project members do you think will be there?

A: I'm expecting Jan, Alex, Claudia, and David, and maybe Olga and Andrei.

b.
```
<dialogueAct xml:id="da1" target="#m1" sender="#b" addressee="#a" dimension="task"
      communicativeFunction="setQuestion"/>
<dialogueAct xml:id="da2" target="#m2" sender="#a" addressee="#b" dimension="task"
      communicativeFunction="answer" functionalDependence="#da1"/>
```

The property of 'responsiveness' is related to what in the literature is called 'backward-looking'. For example, in DAMSL the communicative functions are divided over two categories: forward-looking and backward-looking. Backward-looking functions are defined as those functions that indicate how the current utterance relates to the previous discourse. These include not only answers and other dialogue acts whose semantic content is co-determined by antecedent dialogue acts, but also feedback acts and other acts concerned with speech editing.

Positive and negative feedback-providing acts depend for their interpretation also on what happened earlier in the dialogue, but in a different way. They are concerned with the processing of what was said before - such as its perception or its interpretation. This is illustrated by the examples in (5).

(5) 1. A: The flight on Tuesday would suit me really well.

B: Okay.

2. A: The flight on Tuesday would suit me really well.

B: On Tuesday?

In the first example, B indicates that he/she has correctly understood A's remark; in the second, he/she checks whether he/she heard (or remembers) correctly what A said. This relation between a positive or negative feedback act and its 'antecedent' is called a *feedback dependence relation.*

A feedback dependence relation indicates one or more preceding *dialogue acts* if the feedback concerns high-level processing, such as understanding, and it indicates a *dialogue segment* in the case of low-level processing, such as hearing what was said. In the latter case, ISO 24617-2:2012 stipulated 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. Similarly, for feedback-eliciting acts and for dialogue acts in the Own Communication Management (OCM) dimension or in the Partner Communication Management (PCM) dimension. In particular, Self-Corrections and Partner Corrections frequently refer to a single word or phrase which does not form a functional segment. To make more accurate annotation possible, this second edition introduces a ‘reference segment’, as a stretch of communicative behaviour that is the object of a feedback dependence relation and that is not a functional segment.

5.3 Rhetorical relations

The possibility of annotating rhetorical relations between dialogue acts in ISO 24617-2:2012 was limited in three respects:

a) no particular set of relations was specified;

b) there was no possibility to indicate the roles of the arguments;

c) it was not possible to distinguish between relations at the level of dialogue acts and relations at the level of their semantic contents.

Since the publication of this standard, ISO 24617-8:2016 (DR-core) was published, which defines an annotation scheme for rhetorical relations. This second edition provides an option for annotating rhetorical relations in dialogue in a more fine-grained manner by importing concepts of the DR-core annotation scheme. See 6.3.

Dialogue acts may also be semantically and pragmatically related through other relations, known as *rhetorical relations* or *discourse relations*, as in the examples shown in (6).

(6) 1. A: It ties you on in terms of the technology and the complexity that you want.

2. A: like for example voice recognition.

3. A: because you might need to power a microphone and other things.

4. A: So that’s one constraint there.

In this example[1], a sequence of four functional segments is contributed by the same participant. The segments in lines 2-4 are all related to the dialogue act expressed in the first segment. Segment 2 is related to the initial statement through an *Exemplification* relation, segment 3 through a *Cause* relation, and segment 4 through a *Restatement* relation.

In view of the lack of a general consensus on the rhetorical relations that should be distinguished and how (see e.g. References [11], [25] and [51]), the first edition of this document did not propose a specific set of relations but just offered a provision for specifying a rhetorical relation. In DiAML-XML this provision plays out in the use of an element called ‘<rhetoricalLink>’ which has attributes referring to two dialogue acts and an attribute for specifying a rhetorical relation. Example (7) illustrates this for indicating a causal relation between two dialogue acts.

1) From the AMI corpus, see Reference [61].

(7a) A: Have you seen Pete today?

B: He didn't come in; he has the flu.

(7b)
```
<dialogueAct xml:id="da1" target="#fs1" sender="#a" addressee="#b"
      dimension="task" communicativeFunction="propositionalQuestion"/>
<dialogueAct xml:id="da2" target="#fs2" sender="#b" addressee="#a"
      dimension="task" communicativeFunction="answer" functionalDependence="#da1"/>
<dialogueAct xml:id="da3" target="#fs3" sender="#b" addressee="#a"
      dimension="task" communicativeFunction="inform"/>
<rhetoricalLink dact="#da3" rhetoAntecedent="#da2" rhetoRel="cause"/>
```

ISO 24617-8:2016 (DR-core) for annotating rhetorical relations defines a set of 'core' rhetorical relations. These relations have been used in DiAML-XML annotations as values of the @rhetoRel attribute in several annotation efforts[24],[43],[44]. The <rhetoricalLink> element was found to be rather coarse-grained, however, for two reasons: (1) it is not possible to indicate the roles of the arguments; and (2) it is not possible to distinguish between relations at the level of dialogue acts and relations at the level of their semantic contents. For example, the annotation in (7b) expresses the existence of a causal relation involving two dialogue acts, but does not say which is the reason and which is the result, nor if and how their semantic contents are involved.

The distinction between using a rhetorical relation to relate the semantic content of two dialogue acts and to relate one dialogue act as a whole to another or to the semantic content of another act, is often described as using a 'semantic' and a 'pragmatic' variant of the relation. The distinction is illustrated by (7a) and (8). Where in (7a) having the flu is the cause of not coming in ('semantic cause'), in (8) beating his children is not the cause of Jim being an idiot, but is a reason for the speaker *to say* that Jim is an idiot, so the causal relation is between the semantic content of the second dialogue act and the performance of the first ('pragmatic cause').

(8) Jim is an idiot. He beats his children.

Using devices from DR-core, the bottom line of (7b) can be replaced by the more fine-grained representation in (7c), which specifies argument roles.

(7c)
```
<drLink arg1="#da2" arg2="#da3" rel="cause">
      <argRole arg="#da2" role="result"/>"
      <argRole arg="#da3" role="reason"/>"
</drLink>
```

The involvement of the semantic contents of the dialogue acts can be indicated in annotations by introducing an attribute (say '@relVariant') with values like 'semantic' and 'pragmatic', but such annotation would not be semantically interpretable since no semantic content is available in the DiAML semantics. Adding content information to DiAML annotations can be achieved by means of 'plug-ins', which are introduced in 9.3. Annex D describes alternative possible plug-ins for semantic content, and a plug-in for rhetorical relations on top of a content plug-in.

In this document the constructs <drLink> and <argRole> are introduced in the DiAML-XML concrete syntax, and the conceptual structures that they encode are added to the DiAML abstract syntax with their semantics. Semantically the <drLink> structure is similar to the <rhetoricalLink> element; the interpretation consists of an update operation that inserts the semantic relation in an addressee's information state, adding a specification of the argument roles. Both link structures form an optional part of the annotation scheme (Type I, semantic optionality): they can be omitted from annotation structures without affecting their well-formedness; their presence just gives additional information. They are both to be used in combination with a plug-in that specifies a set of rhetorical relations and, for using the <drLInk> structure, also a set of argument roles. While the <drLink> structure allows more fine-grained annotations than <rhetoricalLink> structures, the latter possibility remains an option that can be used in situations where fine-grained annotation of rhetorical relations is not considered important. Like the first edition, this second edition of this document does not define any particular set of rhetorical relations. Annex D specifies a plug-in with a set of relations based on DR-core and indicates how other sets of rhetorical relations can be introduced.

5.4 Qualifiers

Dialogue participants may be uncertain about the information they provide, or about their commitment to perform an action. They may also express a certain sentiment about the information or the event that is discussed, or express a reservation in the form of a condition. For the annotation of conditions, uncertainty, and sentiment, so-called *qualifiers* are used; see 7.5.

5.5 Metamodel

The above characterization of the notion of a dialogue act makes use of the following key concepts, which form the backbone of the metamodel for dialogue act annotation in Figure 1:

— sender and addressee;

— participant in another role, or 'side-participant' (not necessarily present);

— markable functional segments and reference segments;

— dialogue act, communicative function, and dimension (semantic content category);

— communicative function qualifier (syntactically optional);

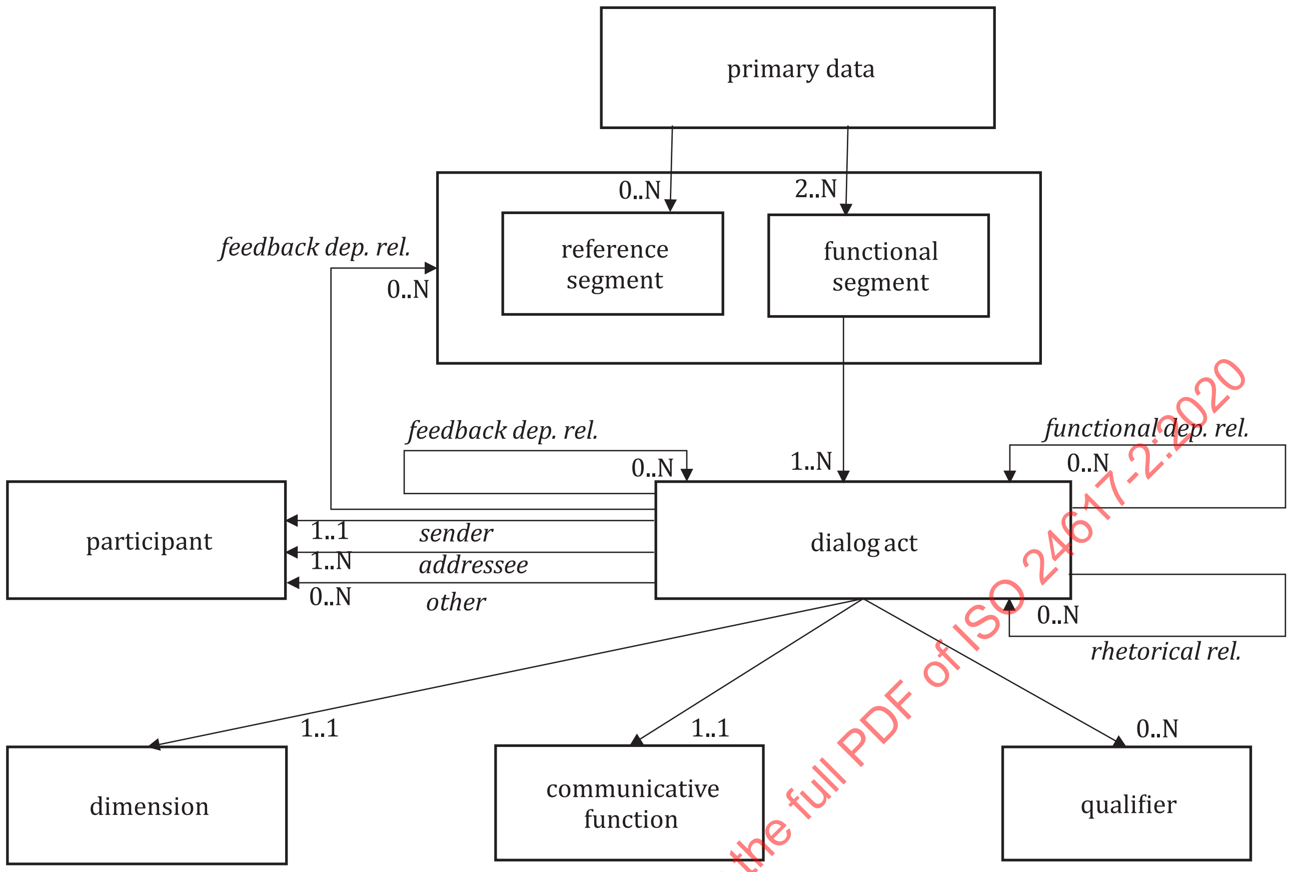
— functional dependence relation and feedback dependence relation (only for certain communicative functions);

— rhetorical relation (semantically optional).

6 Multifunctionality, multidimensionality and segmentation

6.1 Multifunctionality

Participation in a dialogue involves several activities beyond those strictly related to performing the task or activity for which the dialogue is instrumental. In natural conversation, the participants constantly evaluate the perception, understanding and reaction to each other's intentions; the take turns, and the manage the use of time[4],[17]. Communication is thus a multi-faceted activity, and this is reflected in the multifunctionality that dialogue utterances often exhibit. Longer utterances are often *sequentially* multifunctional, i.e., it has parts which each have a different communicative function. A's utterance in example (9) that spans the entire turn unit, for example, is a sequence of five functional segments, with communicative functions such as *feedback giving, request, request, statement,* and *response elicitation.*



(9) A: Yes! Come tomorrow. Go to the church. Bill will be there. OK?

B: The church, OK.

Figure 1 — Metamodel for dialogue act annotation

The occurrence of sequential multifunctionality depends on the way in which a dialogue is segmented, and disappears when sufficiently small segments are used as markables. *Simultaneous* multifunctionality, by contrast, persists even when minimal segments are used as markables. The following example illustrates this.

(10) 1. A: Do you know what date it is?

2. B: Today is the fifteenth.

3. A: Thank you.

A's last utterance in (10) has a thanking function, and can be taken to imply that A has understood and accepted the information provided by B - i.e., as having additionally a positive feedback function. This is a conversational implicature, i.e., a contextually plausible inference; *"Thank you"* does not *necessarily* express positive feedback; it can also be used as a polite way to end an unsuccessful dialogue.

The implication relation between thanking and positive feedback is different from that between a confirmation and a positive answer, where the relation is one of *entailment*, i.e., an implication which is logically valid. Entailment relations occur when one communicative function is a special case of another. Entailed communicative functions should not be considered as instances of multifunctionality.

Turn-initial hesitations are a source of multifunctionality, as the following dialogue fragment illustrates.

(11) 1. A: Is that your opinion too, Bert?

2. B: Uh,.. well,... I guess so.

In this example, A asks a question to B and assigns the turn to B. B accepts the turn and performs a stalling act, buying some time for deciding what to say. So the segment *"Uh,.. well,..."* has both a stalling function and a turn-accepting function. Stalling and turn-accepting are two functions that often go together, but many other combinations of communicative functions cannot co-occur. For example, a functional segment cannot have both a request-accepting and a request-refusing function; these functions are mutually exclusive. The structure of the annotation schema in this document reflects the possible multifunctionality of functional segments.

6.2 Multidimensionality and dimensions

Dialogue act annotation schemes often make use of a clustering of communicative functions that is determined by the intuitive relatedness of certain functions or by similarities in the way they are expressed. An early version of the DIT schema, for example, has a cluster of 'information-seeking functions' for a range of question types, and a cluster of 'information-providing' functions for various kinds of informs and answers[12].

The DAMSL annotation schema[3] is organised into 'layers' and 'dimensions'. Four layers are distinguished: Communicative Status, Information Level, and Forward Looking and Backward Looking Communicative Functions (FLF and BLF); the latter two are clusters of communicative functions; the tags in the other layers are concerned with different kinds of information. The FLF cluster is subdivided into five clusters, including the classes of commissive and directive functions, well known from speech act theory. The BLF cluster has four subclasses: Agreement, Understanding, Answer, and Information Relation. These nine subclasses are referred to as 'dimensions'[31].

An empirical study of multifunctionality[46] mentions six aspects of utterance function as relevant for choosing dimensions: (1) the traditional clustering of illocutionary forces in speech act theory into Representatives, Commissives, Directives, Expressives and Declarations; (2) turn management; (3) adjacency pairs; (4) topical organization in conversation; (5) politeness functions; and (6) rhetorical roles.

A dialogue act tag set can be structured according to multiple dimensions by basing the notion of 'dimension' on the observation that dialogue participants share information not only about the task they pursue but also about the processing of each other's messages, about the allocation of turns, about contact and attention, about the use of time, and about various other aspects of the interaction. They thus perform communicative activities of various types, such as giving and eliciting feedback, taking turns, stalling for time, establishing contact, and showing attention, in addition to those for advancing a certain task or performing a certain activity[14]. Each of these types of activity is concerned with a different category of information. In this document the term 'dimension' is used to refer to these categories and the corresponding activities. This leads to dimensions such as *auto-feedback, turn management, time management*, and *contact management*, besides the dimension formed by the task that motivates the dialogue. Subclause 7.2 describes the dimensions defined in this document.

Not every grouping of communicative functions qualifies as a dimension in the sense of this document. For example, the group of information-giving acts (statements, warnings, answers, corrections, and so on) does not form a dimension since they are not concerned with a particular category of information: information can be given about *any* aspect of the dialogue, such as the underlying task, the understanding of an utterance, a change in topic, or the need to have a break. The same is true of *information-seeking acts* (open questions, check questions, menu questions, and so on), and of the commissive and directive acts (offer, promise, request, suggest, instruct, and so on), which can be about any kind of action. Since these functions can be combined with any kind of information or action, they are called *general-purpose communicative functions*. When combined with a semantic content of a certain category, they form a dialogue act in the dimension corresponding to that category. See 7.3.2.

In contrast with the general-purpose functions, some functions can be used only to address a specific dimension, such as *Turn Keep* and *Turn Release,* which are specific for the dimension of Turn Management; and *Stalling* and *Pause* for the dimension of Time Management. See 7.3.3.

6.3 Segmentation

The multifunctionality of dialogue behaviour is optimally accounted for when communicative functions are assigned to all those segments of behaviour that expresses a dialogue act. Such segments are called *functional segments*, defined more precisely as a *minimal stretch of communicative behaviour that has a communicative function* (possibly more than one, see also definition 3.11). The condition of being 'minimal' ensures that functional segments do not include material that does not contribute to the expression of its communicative function(s). A consequence of this definition is that functional segments may be discontinuous, may overlap, may be embedded in another functional segment, may spread over multiple turns, and may contain parts contributed by different speakers.

For example, consider the segmentation of the turn unit contributed by S in (12).

(12) 1. U: What time is the first train to the airport on Sunday morning?

2. S: The first train to the airport on Sunday morning is .... let me see... at 5:45.

This turn unit contains three functional segments: (1) the discontinuous segment "*The first train to the airport on Sunday morning is at 5:45*", which expresses an answer to U's question in the Task dimension; (2) the embedded segment "*The first train to the airport on Sunday morning*", which provides positive feedback by displaying S's recognition of what U said; and (3) the segment "*let me see*", which has the function of stalling for time. The identification of these functional segments can be viewed as segmenting the turn unit in each dimension in which parts of it have a communicative function; see (13).

(13)	*Dimension*	*Segmentation*
	Task	**The first train to the airport on Sunday morning is** [... let me see...] **at 5:45**
	Auto-Feedback	**The first train to the airport on Sunday morning** / is ... let me see... at 5:45
	Time Management	The first train to the airport on Sunday morning is / **.. .let me see...** / at 5:45

Example (14) illustrates the possibility of a dialogue act to spread over multiple turns. A asks a question, the answer to which consists of a list of items which B communicates one by one.

(14) A: Could you tell me what departure times there are for flights to Frankfurt on Saturday?

B: Certainly. There's a Lufthansa flight in the morning leaving at 08:15,

A: yes,

B: and a KLM flight at 08:50,

A: yes,

B: and a Garuda flight at 10:30,

A: yes,

B: ...

Segments of verbal behaviour have a natural delineation in terms of their constituent words. Nonverbal forms of communicative behaviour, such as hand gestures, head gestures, and facial expressions, do have their own morphology (see e.g. Reference [36]), which can be used to identify nonverbal functional segments. The definition of a functional segment as "minimal stretch of communicative behaviour that has a communicative function" thus applies not only to verbal behaviour but also to nonverbal and multimodal communicative behaviour[42].

In multimodal dialogue, participants combine the use of different modalities including speech and nonverbal elements to form multimodal segments of behaviour which have a communicative meaning. In such situations a functional segment has several modality-specific components, such as a stretch of speech, a facial expression, and a head gesture. See Annex E for examples.

7 Specification of the annotation scheme

7.1 Overview

Following the methodological standard ISO 24617-6, this specification consists of four parts:

a) a metamodel, providing a schematic overview of the concepts that may occur in annotations, and the relations between them;

b) an abstract syntax, providing a formal specification of the inventory of the concepts from which annotations are built up and of the possible ways of combining them, using set-theoretical operations, to form conceptual structures called ‘annotation structures’;

c) a concrete syntax, defining a representation format for annotation structures;

d) a semantics, defining an interpretation of annotation structures (and their representations).

The metamodel is introduced in 5.5. Subclauses 7.2 to 7.5 describe in some detail the main ingredients of the metamodel: the dimensions (7.2), the communicative functions (7.3), the functional and feedback dependence relations (7.4) and the qualifiers (7.5). Clause 8 specifies the abstract and concrete syntax of annotations and their semantics. Together, these components define the markup language DiAML. The concrete syntax specifies a reference representation format, using XML (‘DiAML-XML’); other representation formats have also been used in applications of the annotation scheme, and are equally valid as long as they define a complete and unambiguous representation system[2]. DiAML-XML is syntactically just a compact form of XML; its importance is that it defines a class of XML expressions that have a formal semantics.

7.2 Dimensions

7.2.1 Overview

Not every grouping of related communicative functions makes a dimension in the sense of this document, see 6.2. The most important criteria that a set of proper dimensions for dialogue act annotation should satisfy are the following[39],[40].

a) Empirical validity: each dimension corresponds to a distinct class of well-studied communicative activities that dialogue participants perform, such as turn taking, contact management, and feedback.

b) Orthogonality, or *independence:* each dimension can be addressed by dialogue acts independent of other dimensions. More precisely, for every dimension *D* there should be forms of communicative behaviour which express a dialogue act that is concerned with the activities and the kind of information characteristic for *D*, without necessarily also expressing a dialogue act in one of the other dimensions.

c) Recognisability: each dimension should be recognisable by human annotators as well as by automatic understanding and annotation systems.

A survey of dimensions proposed in the literature, supplemented by statistical and machine learning tests of orthogonality and recognisability[39] shows that the following eleven candidates qualify as dimensions for dialogue act annotation.

2) A representation format is complete and unambiguous for a given abstract syntax if it defines a representation for each annotation structure defined by the abstract syntax, and if each representation unambiguously represents one such annotation structure. For discussion and applications see Reference [16]. For alternative tabular representation formats, see Reference [24].

7.2.2 Task and Task Management

Dialogues are usually motivated by goals, tasks, or activities which are non-communicative in nature, such as obtaining certain information, solving a problem, improving relationships, participating in a game, and so on. The Task dimension is formed by those dialogue acts that are intended to advance the underlying task or activity.

In situations where the dialogue participants are dealing with a task/activity that is not entirely clear to one of them, or where there are restrictions on what one of the participants may say (and when), it commonly happens that the task/activity itself is the topic of dialogue acts, especially at the beginning of a dialogue. Such dialogue acts can be said to form a separate dimension, which in DAMSL has been called 'Task Management'. An example of the first kind of situation is the participation in a quiz, where the quiz master explains the rules of the game[43]; examples of the second kind are interaction in a courtroom and doctor-patient interaction[45].

Task Management has been added as a dimension in this second edition, without introducing communicative functions specific for this dimension. All communicative functions that may be used in the Task dimension may also be used in the Task Management dimension. For certain task domains it may be useful to introduce dimension-specific task management functions, just like it may be useful to introduce specific functions for the Task dimension. This can be realized in the form of a plug-in, as described in 9.3 and Annex D.

7.2.3 Auto-Feedback and Allo-Feedback

The term 'feedback' in dialogue is used to refer to the signalling of understanding, perception, and evaluation of what was said. Feedback morphemes and mechanisms, whether they occur as a single utterance or as a part of a larger utterance, are probably the most important cohesion device in spoken language[5]. Dialogue participants may address several levels of processing of the partner's previous utterances, taking each other into cognitive consideration and showing readiness to communicate, giving attention and receptiveness, recognition, interest and responsiveness to the partner's contributions[10]. Thus, feedback may be reported on various levels, such as *attention, perception, interpretation (understanding), evaluation*, and *execution*[7],[13],[29].

Dialogue participants also monitor the attention, perception, understanding and evaluation of the addressees, and pose themselves such questions as: *Is the addressee paying attention? Does the addressee seem to understand what I mean?* When appropriate, speakers confirm or correct an addressee's processing, or elicit information about it. The terms 'auto-feedback' and 'allo-feedback', borrowed from DIT++, are used to distinguish between the discussion of the speaker's own processing of what was said and that of the addressee's processing. Examples of utterances expressing allo-feedback acts are: *"You see?"* and *"You got me wrong"*.

7.2.4 Turn Management

Turn Management is the distribution of the right to occupy the speaker role, also called the 'floor'[51]. This is rather a normative notion than a behavioural unit[3]. In dialogues with two or three participants, normally only one participant is speaking at any given moment, while the other participants express their involvement through backchannels (like *"uh-huh"*) and nonverbal activity. (Backchannels and nonverbal dialogue acts do not require the sender to occupy the speaker role.) In multi-party dialogue one may find multiple simultaneous speakers[27], and the conversation may effectively split up into sub-conversations involving subgroups of participants.

7.2.5 Time Management

Fluent speech is relatively rare in spontaneous conversation. Disfluent speech production commonly gives rise to issues of timing: at all levels of planning and processing involved in speech production, from retrieving a word to deciding what to talk about next, speakers may experience difficulties which

3) In this document, the corresponding behavioural unit is called a 'turn unit'; see definition 3.24.

give rise to delays [30]. These delays can be minor, giving rise to *stalling* acts, or prolonged, when the speaker performs a *pausing* act to suspend the dialogue for a while.

7.2.6 Discourse Structuring

A dialogue act may indicate the intention to close the discussion of a certain topic, or to move on to a new one, or some other aspect of the speaker's plan for continuing the dialogue. Examples are: *"Something else"* and *"I would like to ask you something"*.

7.2.7 Social Obligations Management

Participating in a dialogue is a social activity, where one is supposed to act in accordance with norms and conventions for social behaviour. Dialogue participants have ethical tasks and obligations, and perform dialogue acts to fulfil these. Social obligation management (SOM) acts are often not just 'social'; they are also used for making the dialogue more transparent. For example, people greet each other not just in order to be friendly, but also to establish and acknowledge their presence, and they wish each other a good day not only for being nice but also to mark the end of a conversation.

7.2.8 Own- and Partner Communication Management

The term 'Own Communication Management' has been introduced[6],[7] for describing the communicative activity of a speaker relating to the management, planning, and execution of his/her own speech production. This activity is indispensable in the description of spoken dialogue, and is illustrated by the occurrence of speech-editing acts such as (self-)repairs and restarts.

Partner Communication Management is concerned with monitoring the speech production of another speaker, such as providing assistance by completing an utterance that the partner is struggling to complete (*completion*), or correcting (part of) an utterance, believing that a speaking error was made (*correct-misspeaking*).

7.2.9 Contact Management

Contact Management acts are concerned with establishing and monitoring contact between speaker and addressee(s). Such acts are particularly important in situations where the participants do not have visual contact, such as in telephone conversations or internet chats. *"Hello", "Allô",* and *"Moshi moshi"* are examples in English, French, and Japanese.

7.3 Communicative functions

7.3.1 Overview

The various dialogue act annotation schemes that have been proposed share a number of communicative functions which are important in almost any type of dialogue. The term 'core dialogue acts' has been used[55] to refer to those acts that are familiar from traditional speech act theory. These include the commissive and directive act types (*promise, offer, request, propose,...),* the 'reportative' speech acts used for stating facts (*assert, conclude*), and the 'expressive' acts for expressing psychological states (*apologise, thank, congratulate*). In this document the terms 'core dialogue act' and 'core communicative function' are used to refer to the types of dialogue acts that are most commonly found in dialogue and their communicative functions, and that are not specific for a particular task domain. Data categories specifying names and definitions of these communicative functions are part of this standard. These include the most common commissive, directive, and reportative acts known from speech act theory and some of the expressive ones, plus a set of other act types which have not been considered much in speech act theory, like acts for turn taking and time management.

The set of communicative functions defined in this document is based on similar criteria as the choice of dimensions. The criterion of *empirical validity* requires that for every communicative function there exist linguistic or nonverbal means which can be used to indicate this function. A good *coverage* is also an empirical requirement. For example, a consequence of the occurrence of what conversational analysts have called ‘adjacency pairs' is that, if an annotation schema includes one element of such a pair, then it should also contain the other. A thanking act is for instance often responded to by a ‘downplayer', and an annotation schema that contains a tag for encoding thankings should therefore also contain one for downplayers. The criterion of *recognisability* reinforces that of empirical validity, and moreover requires that every communicative function has a precise definition, which clearly distinguishes it from other functions.

It is moreover advantageous if the set of communicative functions has the property of *semantic connectedness*, which says that any two communicative functions that can be used for a given dimension are either mutually exclusive or one is a specialization of the other. A scheme with this property has the advantage that an annotator who has decided that a functional segment has a communicative function in a given dimension *D*, can choose from the set of functions available for *D* the most specific one for which there is sufficient evidence. For example, in (15) B's utterance forms an information-providing act in response to A's check question. Given the set of information-providing functions shown in Figure 2, this means that the choice is between the functions *Inform, Agreement, Disagreement, Correction, Answer, Confirm*, and *Disconfirm*. Of these, the functions *Disagreement, Correction* and *Disconfirm* do not apply since there is nothing adversary in what B says. Of the remaining possibilities, *Inform* and *Agreement* are not optimally specific, since they miss the fact that B is responding to a question. Of the two remaining functions, *Confirm* is more specific than *Answer*, and since the expression *“That's right"* expresses agreement with A's expectation, the appropriate function tag is *Confirm*.

(15) A: And that's the first flight tomorrow, right?

B: That's right.

A multidimensional annotation scheme with orthogonal dimensions and semantically connected sets of communicative functions supports the annotation strategy of marking up segments with the most specific communicative function for which there is sufficient evidence, so that a functional segment has at most as many functions as there are dimensions.

This standard includes only rather small numbers of domain-independent core communicative functions:

- general-purpose functions:
  - 6 information-seeking functions,
  - 7 information-providing functions,
  - 8 commissive functions, and
  - 6 directive functions;
- dimension-specific functions:
  - 2 auto-feedback functions,
  - 3 allo-feedback functions,
  - 2 time management functions,
  - 6 turn management functions,
  - 3 discourse structuring functions,
  - 3 own communication management functions,
  - 2 partner communication management functions,
  - 2 contact management functions, and
  - 13 social obligation management functions.

For the possibility of adding other communicative functions, see Clause 9.

7.3.2 General-purpose functions

The general-purpose functions defined in this standard are concerned with obtaining or providing certain information or discussing (communicative or other) actions. The action-discussion functions fall apart into those where the speaker commits himself/herself to perform certain actions (*commissive* functions), and those where the speaker aims to make the addressee(s) perform certain actions (*directive* functions); see Figure 2.

The functions in the information-seeking class are questions of various kinds. Many annotation schemes distinguish several types of question, depending on the type of information that the speaker is looking for and on the speaker's expectations regarding the answer that he/she will get. These distinctions are supported in many languages in the distinction of different sentence types. In this standard, a distinction is made between *propositional questions*, where the speaker wants to know the truth of a given proposition (also known as 'yes/no questions'); *check questions*, which are propositional questions where the speaker expects the answer to be positive; *set questions*, where the speaker wants to know which elements of a given set of entities have a certain property (also known as 'WH-questions'); and *choice questions* (also known as 'multiple-choice questions', 'menu questions', or 'alternatives-questions'), where the speaker wants to know which one of a list of alternatives applies. Special question types are so-called 'test questions' (or 'exam questions'), where the speaker wants to know whether the addressee knows the answer, which the speaker knows. Rhetorical questions only look like questions, and are not included in the class of question types.

The most obvious case of an *information-providing* function is the *Inform*, which in various annotation schemes also goes by the names *statement* and *assertion*, and which is the function of a dialogue act where the speaker has the aim to bring certain information to the addressee's attention. More specific functions are *Agreement* and *Disagreement*, where the speaker believes that the addressee agrees or disagrees, respectively, with the information that is provided, and the *Answer* function, where the speaker provides solicited information. In response to a check question, the speaker may either *Confirm* or *Disconfirm* the addressee's expectation.

Important *commissive* functions are *Promise* and *Offer*, which have in common that the speaker is prepared to commit himself/herself to performing a certain action; the difference is that in the case of a promise this commitment is unconditional, whereas in the case of an offer the commitment occurs only if the addressee accepts the offer.

The prototypical case of a *directive* function is the *Instruct*, where the speaker puts pressure on the addressee to perform a certain action. The *Request* is a conditional directive, which puts pressure on an addressee to perform the requested action if he/she agrees to do so. Note that accepting a request or a suggestion is itself a commissive act, and accepting an offer is a directive act.

While accepting a request implies a commitment to perform the requested action, declining a request can be viewed as a commitment to *not* perform the requested action, and is therefore also a commissive act. Accepting and declining a request are two extremes on a scale of possible responses to a request. The communicative function *Address Request* covers all forms of dealing with a request, with *Accept Request* and *Decline Request* as special cases. Similarly for *Address Offer* and *Address Suggestion*.

The mother-daughter relation in this taxonomy reflects increasing specialization going from mother to daughter; sisters in the taxonomy are mutually exclusive alternatives. The fact that the set of general-purpose functions forms a tree structure shows their semantic connectedness, and can be exploited in annotation processes by using the structure as a decision tree; see Annex E. The general-purpose functions can be used to build a dialogue act in any dimension by combining the function with a semantic content of the category of that dimension. The definitions of the core general-purpose functions are provided in Annex C in the form of data categories conforming to ISO 12620.

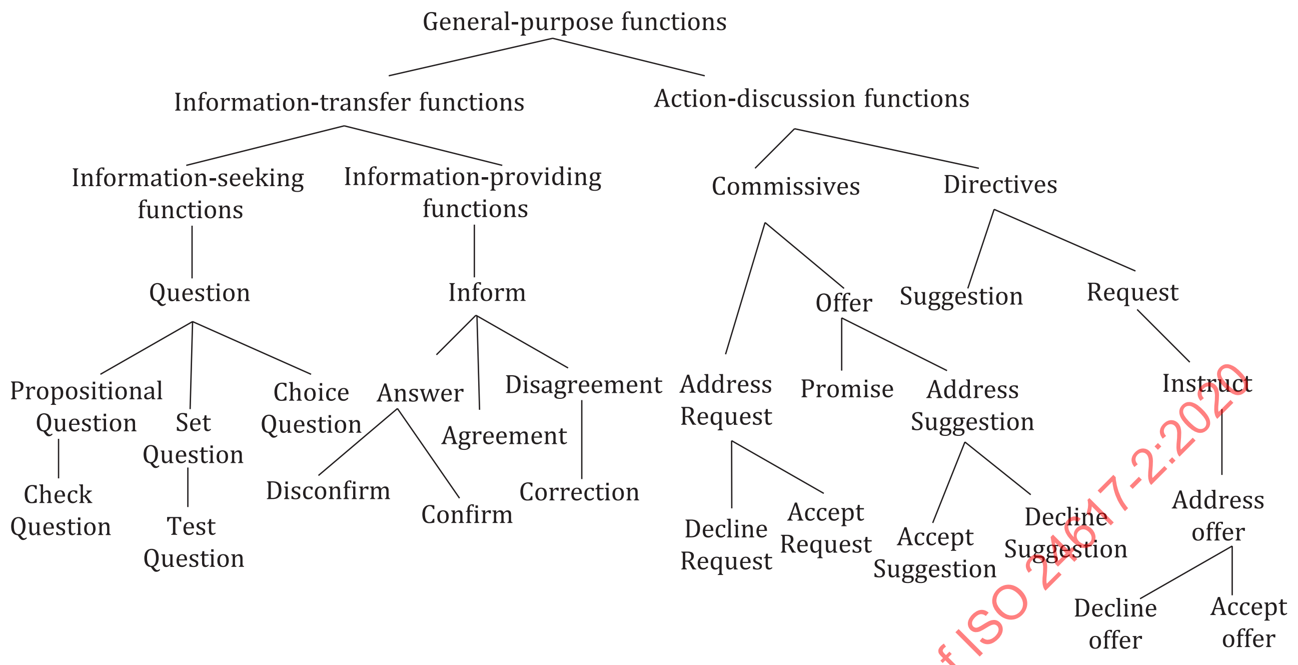


Figure 2 — General-purpose communicative functions

7.3.3 Dimension-specific functions

7.3.3.1 Functions in the Task dimension

Dimension-specific communicative functions for the Task dimension are specific for communication about a particular task domain. For example, specialised communicative functions such as "accept date" and "suggest_exclude_location" have been proposed for a task domain concerned with appointment scheduling. In view of its domain-independence, the present standard does not include any such functions. However, a mechanism to introduce domain-specific communicative functions is described in Annex E.

7.3.3.2 Functions in the Feedback dimensions

Auto- and allo-feedback acts are often performed nonverbally, for instance by nodding, by looking at the speaker (indicating attention), by cupping a hand behind an ear ("I didn't hear you"), or by blinking.

Feedback-providing acts can be divided into positive and negative ones. In the Auto-feedback dimension, positive acts signal that the sender successfully processed a previous utterance; negative ones that a processing problem was encountered. In the Allo-feedback dimension, positive acts signal that the sender believes that the addressee processed a previous utterance successfully; negative ones that the sender believes that the addressee was unsuccessful. Feedback elicitation acts express that the speaker wants to know whether the addressee was successful in processing one or more previous utterances.

Some annotation schemes distinguish various levels of processing to which feedback acts may refer; such as the DIT++ scheme[15]. Annex D describes a mechanism for importing the fine-grained feedback functions from DIT++ in DiAML annotations.

7.3.3.3 Functions in the Turn Management dimension

The turn management functions in this standard are defined as the activities that a dialogue participant undertakes for obtaining, maintaining, or giving up the speaker role. Turn management functions can be divided into *turn-initial* ones, which only occur at the beginning of a turn and which are concerned with obtaining the speaker role, and *turn-final* ones, which occur only within or at the end of a turn end which are concerned with keeping the speaker role or making it available. A functional segment may thus have both a turn-initial and a turn-final turn management function.

7.3.3.4 Functions in the Time Management dimension

Stalling for time is a widespread phenomenon in spoken interaction and may occur for a variety of reasons. It is typically indicated by slowing down and using fillers like *"uh", "let me see", "you know", "well".* Fillers and slowing down can be used when the speaker needs just a few seconds (rather than several minutes or even more). The communicative function characterizing this behaviour is called '*stalling*'. A speaker who needs more time than just a few seconds, for instance to look up something, or because he/she is interrupted by something urgent, should do something else. This is where expressions like *"just a minute", "hold on", "momentito", "un instant", "veuillez patienter"* are used, which signal that the speaker is briefly suspending his participation in the dialogue but intends to resume shortly. This is called '*pausing*'.

7.3.3.5 Functions for discourse structuring

Dialogue participants may structure the interaction explicitly by opening and closing the dialogue, by introducing, changing, or closing a topic, by indicating what they intend to do next, or what they would like another participant to do next. When the discourse structure is addressed explicitly by dialogue acts, this is done most often using a general-purpose function, as in *"Peter, will you introduce the next item?"*

7.3.3.6 Functions in the Own and Partner Communication Management dimensions

Own communication management (OCM) acts occur when a speaker edits his own speech, and most commonly take the form of self-corrections (also called 'repairs') and retractions. The most common forms of Partner Communication Management (PCM) acts are the correction of a partner's speaking error and the completion of an utterance which the partner is struggling to complete.

7.3.3.7 Functions for social obligations management

Of the numerous dialogue acts that can be performed for social functions, in some cultures and languages more than in others, some are found very frequently in all kinds of dialogue. These include greetings and valedictions, at the beginning and end of a dialogue, respectively. Introducing oneself is also common in many interactive situations. Apologies are often used when a dialogue participant has misunderstood another participant, or is unable to fulfil a request or to answer a question. Thanking occurs frequently in those situations where one participant performs a service or provides help, and is also often used to initiate the closing of a dialogue. All these dialogue acts tend to come in initiative-response pairs, such as an initial and a response greeting, an apology and its acceptance, and a thanks and a 'downplayer' (*"De nada"; "Pas de quoi"*). In this document, only SOM functions are defined that seem to occur in every language and culture.

7.3.3.8 Functions for contact management

In certain circumstances it may happen that one is not sure that the person(s) one wanted to talk to is/are actually present and available for communicating with. *"Anybody home?"* and *"Are you still there?"* are much-used expressions to check this, and "*Hello*" when picking up the phone is similarly an indication of one's presence and availability.

7.3.4 Responsive communicative functions

As mentioned in Clause 5, some dialogue acts by the very nature of their communicative function respond to previous dialogue acts, and for the determination of their meaning depend on the semantic content of these previous dialogue acts. The following communicative functions defined in this document are responsive.

— Information-providing functions: Answer, Confirm, Disconfirm, Correction, Agreement, Disagreement.

— Directive functions: Accept Offer, Decline Offer, Address Offer.

— Commissive functions: Accept Request, Decline Request, Address Request, Accept Suggestion, Decline Suggestion, Address Suggestion.

7.4 Functional and feedback dependences

Dialogue acts with a responsive communicative function depend on a previous dialogue act for their semantic content and thus have a 'functional dependence relation', see 5.2. A feedback dependence relation connects a feedback act, an OCM act, and a PCM act to the stretches of preceding communicative behaviour whose processing they are concerned with. The latter characterization leaves open the possibility that a feedback act with a responsive communicative function, such as an answer to a feedback question.

Dialogue acts with a feedback-specific communicative function are expressed by utterances like *"Okay", "Uh-huh", "Yes"* (positive auto-feedback), *"No no no"* (negative allo-feedback), *"Right"* (positive allo-feedback), *"Huh?", "What?"* (negative auto-feedback), and *"Okay*?" (feedback elicitation) - typically with a particular intonation and accompanying facial expressions. Such utterances by themselves contain little semantic information. The same is true for OCM- and PCM-acts with a dimension-specific communicative function. For their semantic content, these acts depend on the stretches of communicative behaviour that they provide or elicit information about. Feedback dependence relations enable this.

Feedback acts with a general-purpose communicative function are either responsive or non-responsive. For a responsive one the semantic content is determined by the semantics of the functional segment that expresses it combined with the semantics of the functional segment(s) of the dialogue act(s) that they functionally depend on; for non-responsive ones the semantic content is locally determined by the functional segment by which they are expressed. Feedback acts with a general-purpose communicative function therefore do not have feedback dependence relations. The same goes for OCM- and PCM-acts. In conclusion, a dialogue act either has no dependence relation at all or it has either a functional or a feedback dependence relation, as determined by its communicative function.

7.5 Qualifiers

A limitation of virtually every dialogue act taxonomy is that it fails to capture subtleties in the performance of communicative actions relating to such phenomena as modality, conditionality, emotions and attitudes. For example, it is customary to distinguish only two possible responses to an offer: acceptance and refusal. An offer may however be responded to in less clear-cut ways, and can for instance be accepted conditionally, as in (16.2a), or declined with uncertainty, as in (16.2b).

(16) 1. A: Can I offer you some coffee?
2. a. B: Only if you have it ready.
b. B: Maybe later

All suggestions and requests can also be accepted and declined conditionally and with uncertainty. Similarly, information-providing acts may express the speaker's awareness that he/she possesses uncertain information, as illustrated in (17).

(17) 1. A: Do you know who'll be coming tonight?
2. B: I have a hunch that Mary won't come.
3. B: Peter, Alice, and Bert will probably come.

Many dialogue acts can also be performed with an expression of the sender's attitude or emotional stance with respect to the semantic content of the act or of toward the addressee, for instance as in (18).

(18) a. A: Can you tell me what time is the first flight tomorrow?
B: The first flight tomorrow morning is at seven-thirty.
A: Perfect!
b. A: What about a fresh cup of coffee?
B: Ah, you're wonderful!

In order to be able to represent such phenomena, this standard includes certain *qualifiers* that may be associated with a communicative function. A corpus-based study[41] indicates that uncertainty and conditionality can be captured by means of binary distinctions (certain/uncertain; conditional/unconditional); the standard therefore defines two binary-valued attributes, @certainty and @conditionality. The certainty values 'uncertain' and 'certain' can be associated with information-providing functions in order to represent the speaker's expression of (un)certainty about the correctness of the information that he/she provides, and with commissive functions to indicate uncertainty about the speaker's commitment to perform the action under discussion. This attribute has the default value 'certain'. The conditionality values 'conditional' and 'unconditional' can be used with action-discussion functions, and concern the ability and willingness of the participant whose action is under discussion. This attribute has the default value 'unconditional'.

For representing a speaker's attitude or emotional stance this document follows the common distinction between 'sentiment' and 'emotion', where the former is understood as a view or opinion that is held (and is often shared by a group), while the latter is a mental state that occurs as a result of arousal by an internal stimulus (a thought) or an external one (an event). Sentiment is often characterized by a polarity (positive or negative), possibly with an intensity, see e.g. Reference [49]. For emotions a wide variety of descriptors has been proposed in the literature, ranging from Ekman's six basic emotions[32] to several hundred possible values and complex structural descriptions. The standard described in this document includes in its first edition the option of annotating a sentiment or an emotion by means of a qualifier as the value of the attribute @sentiment, for which any set of values may be chosen depending on the domain, annotation goals, or interactive setting, or on theoretical preferences. Experiences in using the standard have shown that it may be useful to characterize participants' sentiment with respect to something in the dialogue as 'positive' or 'negative'; in this second edition these values are stipulated for the @sentiment attribute. The annotation of emotions, by contrast, requires more articulate structures. Annex D describes a plug-in mechanism for importing concepts from the W3C recommendation 'EmotionML'[26] in order to enrich DiAML annotations with emotion-related information.

8 The Dialogue Act Markup Language (DiAML)

8.1 Overview

The Dialogue Act Markup Language DiAML has been designed in accordance with ISO 24617-6, which implements the distinction made in the Linguistic Annotation Framework (LAF, ISO 24612) between *annotations* and *representations*. The term 'annotation' refers to the linguistic information that is added to segments of language data, independent of the format in which the information is presented; 'representation' refers to the format in which an annotation is rendered. According to LAF, *annotations* are the proper level of standardization, rather than *representations*. Following ISO 24617-6, this distinction is implemented in the DiAML definition by a syntax specification that defines, besides a class of XML-based *representation structures*, also a class of more abstract *annotation structures*. These specifications are called *concrete* and *abstract syntax*, respectively. Annotation structures are set-theoretical structures, consisting of concepts of the classes that populate the metamodel (see Figure 1). The concrete syntax defines a reference format for rendering annotation structures in XML. Alternative representation formats for DiAML annotation structures are discussed in Reference [24].

The following subclauses outline the abstract syntax, the concrete syntax, and the semantics of DiAML annotations; Annex A contains a formal specification of the abstract syntax and the concrete DiAML-

XML syntax. A detailed specification of the semantics of DiAML annotation structures can be found in Reference [19].

8.2 Abstract syntax

The abstract syntax of DiAML consists of

a) a specification of the elements from which annotation structures are built up, called a 'conceptual inventory', and

b) a specification of the possible ways of combining these elements to form annotation structures.

The conceptual inventory of DiAML consists of sets of dialogue participants, dimensions, communicative functions, functional segments, and qualifiers.

An annotation structure is a set of *entity structures* and *link structures.* Entity structures contain semantic information about a dialogue segment; link structures describe semantic relations between entity structures. Entity structures are always of the general form ⟨m,z⟩, where 'm' is a markable and 'z' designates a structure that describes some linguistic information. Link structures are typically of the form ⟨e1, e2, R⟩, consisting of two entity structures and a relation.

The entity structure of central interest in DiAML is a pair ⟨m,α⟩, of which the linguistic information 'da' is a so-called 'dialogue act structure', which contains the information that characterizes a single dialogue act. This includes minimally a specification of the sender, the addressee(s), and the communicative function. For dialogue acts with a general-purpose communicative function, the dimension of the semantic content is another component; for dialogue acts with a dimension-specific function the dimension does not need to be specified, since it is inherent in the definition of the function. General-purpose functions may have one or more qualifiers. For a dialogue act which depends semantically on (the interpretation of) one or more previous dialogue segments, a sixth component is a set E of elements that the act depends on through functional or feedback dependence relations. In a setting where participants other than the sender and the addressees should be taken into account, an additional element is a set H of 'other participants' (see Figure 1). A dialogue act structure is therefore in the simplest case a triple $\langle S, A, f_d \rangle$, consisting of a sender S, a (set of) addressee(s) A, and a dimension-specific function f_d, and in the most complex case a 7-tuple as in (19), with a general-purpose function f a dimension d, a set q of qualifiers, and a set E of dialogue units that the act depends on.

(19) $\alpha = \langle S, A, H, f, d, q, E \rangle$

A link structure in DiAML is a triple $\langle \varepsilon, E, \rho \rangle$ consisting of an entity structure ε, a set E of one or more entity structures, and a rhetorical relation ρ, which relates the dialogue act in ε to those in E.

8.3 Concrete syntax

The DiAML concrete syntax is defined in accordance with the CASCADES methodology for defining semantic annotation languages, described in Reference [18] and included in ISO 24617-6. This methodology includes the notion of an *ideal representation format*, defined as one which is (1) 'complete' in the sense that every annotation structure defined by the abstract syntax can be represented, and (2) 'unambiguous' in the sense that every representation defined by the concrete syntax represents one and only one annotation structure defined by the abstract syntax. Since the semantics of DiAML is defined for the structures defined by the *abstract* syntax, any two representation formats which are 'ideal' in this sense are semantically equivalent, and every representation in one such format can be converted by a meaning-preserving mapping into any other such format. The DiAML concrete syntax specifies a reference representation format based on XML, called DiAML-XML. This specification lists names of XML tags, attributes, and values corresponding to the various ingredients in the conceptual inventory, and defines the possible ways of combining these elements in XML structures. In particular, XML elements are defined for entity structures and link structures. The dimensions, communicative functions, and qualifiers that can be used in DiAML are defined as data categories, following ISO 12620. Their specification can be found in Annex C.

Entity structures for dialogue acts are represented by an XML element called <dialogueAct>, which has the following attributes:

— @xml:id, whose value is a unique identifier of a dialogue act structure;

— @target, whose value refers to a functional segment;

— @sender, @addressee, and @otherParticipant, whose values refer to dialogue participants, identified in the metadata of the annotated primary data; the attribute otherParticipant is optional;

— @dimension, whose value names one of the dimensions defined in this standard;

— @communicativeFunction, whose value names one of the communicative functions defined in this standard;

— @certainty, @conditionality, and @sentiment, whose values is one of the qualifiers defined in this standard. The attributes are optional;

— @feedbackDependence, whose values refer to one or more dialogue acts or reference segments that the given dialogue act has a feedback dependence relation with; see 7.4;

— @functionalDependence, whose values refer to one or more dialogue acts that a responsive dialogue act has a functional dependence relation with.

Link structures are represented either by the XML element <rhetoricalLink> or by the element <drLink>. The <rhetoricalLink> element has the following attributes:

— @dact, whose value refers to a dialogue act that is rhetorically related to other dialogue acts;

— @rhetoRelatum, whose value refers to one or more dialogue acts that the given dialogue act is rhetorically related to;

— @rhetoRel, whose value names a rhetorical relation.

The <drLink> element has the following attributes:

— @arg1 and @arg2, whose values refer to two rhetorically related dialogue acts or their semantic content, if a plug-in for semantic content is used (see Clause 9 and Annex D);

— @rel, whose value is a rhetorical relation;

and makes use of embedded <argRole> elements, which have an attribute @arg, whose value identifies a dialogue act, and an attribute @role, whose value names an argument role.

Example (20c-d), shows the abstract annotation structure and its DiAML-XML representation of the dialogue fragment in (20a), segmented as shown in (20b).

(20a) P1: What time does the next train to Utrecht leave?
P2: The next train to Utrecht leaves I think at 8:32.

Annotations may be attached to primary dialogue data in a variety of ways; they may be attached directly to stretches of speech, defined by temporal begin- and end-points, or to structures at lower levels of description, such as the output of a tokenizer. It is assumed that functional segments are identified at another level of XML representation, following ISO 24612-1. P2's utterance is segmented into two overlapping functional segments: m2 in the Auto-Feedback dimension (reflecting that the repetition of a large part of an utterance signals positive feedback on understanding it) and m3 in the Task dimension. Following the guidelines of the Text Encoding Initiative (see Reference [53]), the prefix '#' is used to indicate that the prefixed value is identified either in the metadata of the primary data or in another layer of annotation, or elsewhere within the same representation.

Note that the abstract annotation structure in (20c) is a set of three elements, corresponding to the three dialogue acts in this fragment, where the second and the third element both have the first element embedded, indicating their dependence on the first dialogue act.

(20b) *Segmentation of the exchange in (20a)*:
m1 = *What time does the next train to Utrecht leave?* (Task dimension)
m2 = *The next train to Utrecht leaves* (Auto-Feedback dimension)
m3 = *"The next train to Utrecht leaves I think at 8:32."* (Task dimension).

(20c) *Annotation structure according to DiAML abstract syntax*:
{⟨m1,⟨p1,p2,setQuestion,Task⟩⟩,
⟨m2,⟨p2,p1,autoPositive,{⟨m1,⟨p1,p2,setQuestion,Task⟩⟩}⟩⟩,
⟨m3,⟨p2,p1,answer,Task,{uncertain},{⟨m1,⟨p1,p2,setQuestion,Task⟩⟩}⟩⟩}

(20d) *DiAML-XML annotation representation*:

```
<diaml xmlns:"https://www.iso.org/diaml/">
    <dialogueAct xml:id="da1" target="#m1" sender="#p1" addressee="#p2"
        communicativeFunction="setQuestion" dimension="task"/>
    <dialogueAct xml:id="da2" target="#m2" sender="#p2" addressee="#p1"
        communicativeFunction="autoPositive" feedbackDependence="#da1"/>
    <dialogueAct xml:id="da3" target="#m3" sender="#p2" addressee="#p1"
        communicativeFunction="answer" certainty="uncertain" dimension="task"
        functionalDependence="#da1"/>
</diaml>
```

8.4 Semantics

DiAML annotation structures have a semantics in terms of information-state updates. The most important kind of structure defined by the DiAML abstract syntax, the dialogue act structure, is a *functional* characterization of a dialogue act. It does not correspond to a complete dialogue act, since it does not include the semantic content (but only a semantic content category, a 'dimension'). The semantics of a complete dialogue act is obtained by combining the interpretation of a dialogue act structure with a semantic content. This is accomplished by applying the interpretation $I_a(\langle s,\alpha\rangle)$ of an entity structure which contains a dialogue act structure α, to the semantic content $\kappa 1(s)$ of the functional segment that expresses the dialogue act. The result is an information state update operation as shown in (21) for a dialogue act that has no functional dependences.

(21) $I_a(\langle s,\alpha\rangle) = I_a(\alpha)(\kappa 1(s))$

The interpretation $I_a(\alpha)$ of a dialogue act structure α is defined as follows for those structures without qualifiers.

(22) $I_a(\langle S,A,f,d\rangle) = I_a(f)(I_a(S), I_a(A), I_a(d))$

I.e., the interpretation of a dialogue act structure is the interpretation of its communicative function, applied to the interpretations of its sender, its addressee, and its dimension. Annex A provides more information about the DiAML semantics; a full specification can be found in Reference [19].

9 Extension and customization

9.1 Overview

This document is extensible in the following four respects.

a) Dimensions: due to the orthogonality of its dimensions, additional dimensions may be introduced as long as they are orthogonal to the dimensions already present and to each other.

b) Communicative functions: the taxonomy of communicative functions expresses semantic relations between: dominance relations express different degrees of specialization; sister relations express mutual exclusivity. Communicative functions may be added to the taxonomy as long as they respect these relations.

c) Qualifiers: due to the orthogonality of the qualifier attributes, additional orthogonal attributes may be introduced. The possible values of each quantifier are mutually exclusive; additional values may be introduced as long as they respect this property.

d) Rhetorical relations: this document does not specify a set of relations, but recommends the use of the relations defined in ISO 24617-8 or an extension of that set -to be plugged in; see Annex D.

The annotation scheme defined in this standard is more comprehensive than other existing dialogue annotation schemes. For some, relatively simple application domains it may be unnecessarily complex, while on the other hand the domain-independence of the scheme has the effect that it may lack some communicative functions or other concepts that are important for specific applications. The sets of communicative functions, qualifiers, and relations among dialogue units defined in this standard cannot be expected to be all that is needed for every application, for every task domain, for every type of interaction, and for every annotation purpose. The structural properties of the annotation scheme are however useful also for simplifying the annotation scheme for relatively simple applications and for defining extensions to customize the scheme to a certain application, in particular for the use cases UC3 and UC4 (see Clause 4).

Simplified versions of the annotation scheme are easily defined thanks to the structural properties of the scheme, which result from a careful design and the optionality of parts of it; see 9.2.

Extensions of the annotation scheme affect the interoperability of the annotations that result from its application in the use cases UC1 and UC2; whether this is desirable or undesirable depends on the purpose of the annotations. This second edition of this document makes use of *triple-layered annotation scheme plug-ins*, a powerful mechanism for enriching DiAML-annotations. See 9.3.

9.2 Simplifying the annotation scheme: options and selections

The structural properties of the annotation scheme defined in this standard which contribute to its customizability are mainly the following.

a) The dimensions are orthogonal: a functional segment expressing a dialogue act in one dimension does not necessarily have a communicative function in another dimension.

b) Each communicative function has a semantic definition in terms of how it defines an update operation on the information states of dialogue participants when combined with a semantic content.

c) The dimensions are orthogonal: a functional segment expressing a dialogue act in one dimension does not necessarily have a communicative function in another dimension.

d) Each communicative function has a semantic definition in terms of how it defines an update operation on the information states of dialogue participants when combined with a semantic content.

e) The set of general-purpose communicative functions is *semantically connected* (any two functions are either mutually exclusive or one is a specialization of the other). The general-purpose communicative functions are semantically connected, and for each dimension the set of dimension-specific communicative functions is semantically connected.

f) Due to the semantic connectedness property, a functional segment never needs to be annotated with more than one communicative function in each dimension where it has a function. A functional segment thus has maximally as many functions as there are dimensions.

Simplified subschemes of the annotation scheme defined in this document can be defined relatively easily, by leaving out certain ingredients in the following ways.

— By virtue of the orthogonality of the dimensions, any dimension and the corresponding set of dimension-specific communicative functions may be left out of consideration.

— Communicative functions for which there is a less specific function in the annotation scheme may be left out, since the remaining set of functions is still semantically connected.

— Due to their optionality, there's no obligation to use qualifiers and rhetorical relations.

9.3 Extending the annotation scheme: triple-layered plug-ins and interfaces

In software, a 'plug-in' is an addition to an existing computer program that adds some feature to it, typically in order to customize it. Similarly, annotation tools often come with 'plug-ins' in order to customize them for a particular annotation scheme. Annotation schemes sometimes come with 'plug-ins' in the sense of a vocabulary of terms that may be used in annotations; this is for example the case for the EmotionML annotation scheme[26]. This subclause describes the notions of a *triple-layered annotation plug-in* and a *plug-in interface*, as introduced in Reference [21].

According to the SemAF principles of semantic annotation, as laid down in ISO 24617-6, a semantic annotation scheme has a three-part definition, consisting of (1) an abstract syntax that specifies the well-formed '*annotation structures*' as set-theoretical constructs, such as pairs and triples; (2) a semantics that specifies the meanings of the annotation structures; (3) a concrete syntax that specifies a representation format for annotation structures. Formally, the definition of an annotation scheme is thus a triple $L_a = \langle AS_a, CS_a, Sm_a \rangle$, formed by specifications of an abstract syntax (AS_a), a concrete syntax (CS_a), and a semantics (Sm_a). Each of these components is further structured:

— the abstract syntax specification AS_a is a pair $\langle CI_a, AC_a \rangle$, consisting of the conceptual inventory (CI_a) and the specification of entity structures and link structures (AC_a); together, these define the class of well-formed annotation structures;

— the concrete syntax specification CS_a is a triple $\langle V_a, CC_a, F_a \rangle$, where V_a is a vocabulary, CC_a is the specification of a class of syntactic structures, and F_a is an encoding function that maps AS_a-annotation structures to CS_a-representations. Together, V_a and CC_a define the class of well-formed representations;

— the semantic specification Sm_a is a pair $\langle M_a, I_a \rangle$, consisting of a model and an interpretation function.

Altogether, the definition of an annotation scheme is thus a nested triple:

(23) $$L_a = \langle AS_a, CS_a, Sm_a \rangle$$
$$= \langle \langle CI_a, AC_a \rangle, \langle V_a, CC_a, F_a \rangle, \langle M_a, I_a \rangle \rangle$$

A *triple-layered ('tripartite') annotation plug-in* is an annotation scheme that can be added on to a host annotation scheme, in general requiring a *plug-in interface* for allowing annotations that combine elements from the two schemes (see Reference [21]). Being an add-on annotation scheme, an annotation plug-in has the same tripartite structure, introducing (1) additional entity structures and link structures in the abstract syntax, (2) the semantics of these structures, and (3) their encodings.

An interface for a given host annotation scheme and plug-in defines link structures that relate entity structures of the two schemes. It has again the tripartite structure of an annotation scheme, but it does not introduce new entity structures but only new link structures.

Added on to a host annotation scheme, a plug-in in general defines additions to all three parts of the host scheme [see example (23)]: to the host's concrete syntax but also to the abstract syntax and semantics, making sure that the extended annotation scheme complies with the principles of semantic annotation laid down in ISO 24617-6. A plug-in therefore has the same three-part structure, specifying additional concepts, entity structures and link structures, their concrete encodings, and their semantics. To emphasize their highly structured character, very different from a plug-in in the form of a vocabulary, such plug-ins are referred to as 'triple-layered plug-ins'. In the rest of this document, 'plug-in' is used as short for 'triple-layered plug-in'. A plug-in interface, defining link structures that relate entity structures of the two schemes, is again a structured specification of abstract structures, their concrete encodings, and their semantics, but an interface is typically simpler than a plug-in since it only introduces structures for 'linking' the structures of the host annotation scheme and the plug-in.

Note that plug-ins for the annotation scheme of this standard must respect the structural properties listed in the previous subclause. In particular, any plug-in that introduces new dimensions or communicative functions is required to ensure that the resulting set of dimensions is still orthogonal and that the resulting set of communicative functions is semantically connected.

Annex D describes triple-layered plug-ins and plug-in interfaces for adding semantic content information to dialogue acts, domain-specific communicative functions, rhetorical relations, emotions, more fine-grained feedback functions, and additional functions for social obligations management.

Annex A
(normative)

Formal specification of DiAML

A.1 Overview

Following ISO 24617-6, the formal specification of DiAML is a triple ⟨AS, CS, Sm⟩, consisting of an abstract syntax (AS) that specifies a class of annotation structures as triples and other set-theoretic constructs (made up of the concepts that populate the metamodel, shown in Figure 1), a concrete syntax (CS) that defines a representation format for these structures, and a semantics (Sm) that defines their meaning.

A.2 Abstract syntax

The abstract syntax of DiAML consists of: (a) a specification of the elements from which annotation structures are built up, called a 'conceptual inventory', and (b) a specification of the possible ways of combining these elements and form annotation structures.

a) Conceptual inventory

The conceptual inventory of DiAML consists of the following finite sets:

— *DP:* dialogue participants;

— *Dim:* dimensions;

— *CF:* communicative functions, with a specified subset of responsive communicative functions;

— *M:* markables, divided into *FS:* functional segments and *RS*: reference segments;

— *QV:* a set of finite sets Q_1,.. Q_k, of qualifiers;

— *RR:* rhetorical relations; *AR*: argument roles; and a function ρ from *RR* to *AR*.

b) Annotation structures

An annotation structure is a set of *entity structures* and *link structures.* Entity structures contain semantic information about a functional segment; link structures describe semantic relations between functional segments.

b1. Entity structures. An entity structure is a pair ⟨*m*,*α*⟩ consisting of a functional segment *m* and a 'dialogue act structure' α, which contains information characterizing a dialogue act, expressed by *m.* Formally, a dialogue act structure is a 7-tuple ⟨*S, A, H, d, f, q, D*⟩ specifying a sender (*S*), the addressee(s) (*A*), possibly other participants (*H*), a dimension (*d*), a communicative function (*f*), a set of qualifiers (*q*) (optional), and functional and feedback dependences (*D*) (if any).

b2. Link structures. A link structure is either (1) a triple or quadruple ⟨*ε, E, R, [r]*⟩ consisting of an entity structure (ε), a non-empty set of entity structures (*E*), a rhetorical relation (*R*) that relates the dialogue act in ε to those in *E*, and optionally a rhetorical relation type(*r*), or (2) a nested structure ⟨⟨*ε,ρ1*⟩, ⟨*E, ρ2*⟩, *R, [r]*⟩, with *ε, E, R and r* as before and where *ρ1* and *ρ2* are the argument roles of *R*.

A.3 Concrete syntax

The concrete syntax specified defines an *ideal representation format*[16], i.e., a format that is (1) *complete* in the sense that every annotation structure defined by the abstract syntax has a representation defined by the concrete syntax, and (2) *unambiguous* i.e., every representation defined by the concrete syntax represents one and only one annotation structure defined by the abstract syntax. Since the semantics of DiAML is defined for the structures defined by the *abstract* syntax, any two representation formats which are 'ideal' in this sense are semantically equivalent, and every representation in one such format can be converted by a meaning-preserving mapping into any other such format.[4] The representation format defined is based on XML, and is referred to as DiAML-XML. Alternative representation formats are considered in Reference [24].

The DiAML-XML concrete syntax is formally a triple ⟨V, CC, F_{AC}⟩, consisting of (a) a vocabulary (V), specifying XML names for the elements of the conceptual inventory; (b) a set of XML structures (CC) in which these names can be used; and (c) an encoding function (F) that specifies the XML representation of the annotation structures defined by the abstract syntax (including the assignment of vocabulary items to conceptual inventory items). The encoding function F is a total function, ensuring the completeness of the DiAML-XML format, as is the inverse function F^{-1}, ensuring the unambiguity of the representations.

The DiAML-XML vocabulary contains the following attributes and values for use in entity structure representations by means of the XML element `<dialogueAct>`:

— the obligatory attribute `@xml:id`, whose value is a unique identifier within an annotation representation;

— the obligatory attribute `@target`, whose value refers to a functional segment;

— the obligatory attribute `@sender`, whose value refers to a dialogue participant;

— the obligatory attribute `@addressee`, whose values refer to one or more dialogue participants (if it refers for example to two participants, identified as p1 and p2, then the attribute has the value “#p1 #p2”);

— the optional attribute `@otherParticipant`, whose values refer to one or more dialogue participants;

— the obligatory attribute `@communicativeFunction`, whose value names a communicative function;

— the attribute `@dimension`, which is obligatory for those structures in which the value of the `@communicativeFunction` attribute is a general-purpose function, and which is optional for structures with a dimension-specific communicative function. Its value names one of the dimensions;

— the attribute `@functionalDependence`, which is obligatory for dialogue acts with a responsive communicative function; its values refer to one or more dialogue acts. It does not apply in other cases;

— the attribute `@feedbackDependence`, which is obligatory for dialogue acts with a dimension-specific function in the dimensions Auto-Feedback, Allo-Feedback, Own Communication Management, and Partner Communication Management, and which is not applicable in other cases; its values refer to one or more dialogue acts or reference segments;

— the optional attributes `@certainty`, `@conditionality`, and `@sentiment`, whose values specify a communicative function qualifier.

For use in link structure representations, the DiAML-XML vocabulary contains the following items:

— the attributes `@dact` and `@rhetoRelatum`, obligatory in `<rhetoricalLink>` elements; `@dact` has a `<dialogueAct>` element as its value; `@rhetoRelatum` has one or more `<dialogueAct>`s as value : `@rhetoRel` has a rhetorical relation name as its value;

— the attribute `@rhetoEelType`, optional in `<rhetoricalLink>` and `<drLink>` elements, with the possible values “`@semantic`”, “`@pragmasemantic` ”, and “`@semapragmatic`”;

— the attributes `@arg1`, `@arg2`, and `@rel`, obligatory in `<drLink>` elements; `@arg1` has a `<dialogueAct>` element as its value or the representation of a dialogue act’s semantic content (if a plug-in is present

4) Formal definitions and proofs can be found in Reference [16].

for representing such content; see Annex D); `@arg2` has a list of one or more `<dialogueAct>` elements as its value or the representation of their semantic content; `@rel` has a rhetorical relation name as its value;

— the attributes `@arg` and `@role`, obligatory in `<argRole>` elements embedded in `<drLink>` elements; `@arg` has a `<dialogueAct>` element as its value or the representation of a dialogue act's semantic content (if a plug-in is present for representing such content; see Annex D); `@role` has as its possible values the names of the roles of the `@rel` value in the embedding `<drLink>` element; see Table D.3 in Annex D.

The annotated dialogues in Annex E contain illustrations of the use of both types of link structure representation.

The syntactic constructs of DiAML are the elements `<dialogueAct>`, `<rhetoricalLink>`, `<drLink>`, and `<argRole>`. The encoding function F_{AC} specifies the encoding of dialogue act structures and rhetorical link structures by means of these constructs.

A.4 Semantics

A.4.1 Semantics of dialogue acts and communicative functions

DiAML has a formal semantics defined in terms of information-state updates. The details of such a semantics depend on the implementation of information states. The specification of the DiAML semantics makes no further assumptions than that an information state has various components representing different kinds of information, an assumption that is commonly made in proposals for information state representations (see References [1], [13], [30], [35], [38] and [46]) and that an information state has a part (called the 'pending context') for buffering update information that needs to be checked for consistency before it can be added to the consolidated part of the information state. The details of an information-state update semantics also depend on whether a single addressee is considered or multiple addressees, and on whether only the information states of addressees are considered to be updated by dialogue contributions, or also that of the sender or that of other participants. To simplify matters, only the update of a single addressee's information state is considered in this specification.

The most important structure of DiAML, the dialogue act structure, is a *functional* characterization of a dialogue act. It does not correspond to a full-blown dialogue act, since it does not include a semantic content. The semantics of a full-blown dialogue act is obtained by combining the interpretation of a dialogue act structure with a semantic content. Formally, the interpretation of an entity structure $\langle m,\alpha\rangle$ is a function that is applied to the semantic content. For a dialogue act that has no functional or feedback dependences this takes the form shown in (A.1), where I_{DA} is the interpretation function for full-blown dialogue acts, I_a is the interpretation function for DiAML annotation structures, and $\kappa_1(m)$ is the compositional semantics of the functional segment *m.* (The semantics of dependence relations is considered below).

(A.1) $I_{DA}(\langle s,\alpha\rangle) = I_a(\alpha)(\kappa_1(s))$

The interpretation $I_a(\alpha)$ of a dialogue act structure without qualifiers is computed as follows (the semantics of qualifiers is considered below), where case a. applies to dialogue acts with a general-purpose communicative function (f) and case b. to acts with a dimension-specific function (f_d).

(A.2) a. $I_a(\langle S,A,f,d\rangle) = I_a(f)(I_a(S), I_a(A), I_a(d))$

b. $I_a(\langle S,A,f_d\rangle) = I_a(f_d)(I_a(S), I_a(A))$

A dialogue act structure is thus interpreted as the interpretation of its communicative function, applied to (the interpretations of) its sender, its addressee, and, if present, its dimension. As will become clear below, the result is a function that can be applied to a semantic content, resulting in an information state update operation.

A.4.2 Dialogue acts as update operations

The semantics of a complete annotation structure, consisting of the entity structures $\{e_1,..., e_n\}$ and the link structures $\{L_1,..., L_k\}$, is defined as the successive application of the update functions corresponding to the entity and link structures, ordered by the precedence relations ($<_S$) between their functional segments.

The semantic relatedness of dialogue act types, as visualized in Figure 2, is brought out in their interpretation as information state updates. Compare, for example, a *Confirm* act and an *Answer*. According to their definitions (in Annex C) an *Answer* with semantic content *p* tells an addressee (A) three things: (1) that the sender (S) wants to make the information *p* available to A; (2) that S believes that A wants to know whether *p,* and (3) that S assumes that *p* is true. A *Confirm* act with the same content additionally tells A that (4) S believes that A already thought that *p,* but was uncertain about it. Relations like the one between *Confirm* and *Answer* are captured in the DiAML semantics by defining the interpretation of a communicative function as a combination of *elementary update functions*, each of which takes care of a single update item. The semantics of the *Answer* function is the combination of three elementary update functions, and that of the *Confirm* function is the combination of these three update functions and a fourth one, which expresses the difference between *Confirm* and *Answer*.

Elementary update functions are defined as parameterised schemes with parameters for a sender, an addressee, and an information state component, such as the following ones.

(A.3) $U_{10}(X,Y,D_i,p)$: add to component D_i of *Y*'s pending context the information that participant *X* wants to know whether $p_;$

$U_{11}(X,Y,D_i,p)$: add to component D_i of *Y*'s pending context the information that participant *X* assumes participant *Y* to know whether *p*.

These two schemes can be used to specify the semantics of the communicative function *Propositional Question.*

(A.4) $I_a(\text{Propositional Question}) = \lambda X.\ \lambda Y.\ \lambda D_i.\ \lambda z.\ U_{10}(X,Y,D_i,z) \cup U_{11}(X,Y,D_i,z)$

Applied to two participants *Sys* and *Usr* and a task-related question, the function specified in the right-hand side of (A.4) produces the update function in (A.5), where Sys_{TaskC} denotes the component of the system's pending context where task-related information is buffered.

(A.5) $I_a(\textit{Prop. Question})(Usr, Sys, TaskC) = \lambda p.\ U_{10}(Usr, Sys, Sys_{TaskC}, p) \cup U_{11}(Usr, Sys, Sys_{TaskC}, p)$

Applied to a value for *p*, this specifies how to update *Sys*'s pending context. For example, when *Usr* asks *Sys* whether flight KLM flight 476 departs at 19:15, formalized as Dep(KL476)=9:15, then if *Sys* understands *Usr* correctly, the component Sys'_{TaskC} of the system's pending context is extended with two beliefs:

a) according to the update U_{10}(*Usr, Sys, TaskC*, Dep(KL476)=19:15), *Sys* believes that *Usr* wants to know whether Dep(KL476) is 19:15;

b) according to the update U_{11}(*Usr, Sys, TaskC*, Dep(KL476)=19:15), *Sys* believes that *Usr* assumes that *Sys* knows whether Dep(KL476) is 19:15.

A.4.3 Qualifiers and relations

A.4.3.1 Qualifiers

Qualifiers make the information state updates of communicative functions more specific. They come in two varieties, 'restrictive' and 'additive' ones[19]. Restrictive qualifiers make the preconditions of a communicative function more specific, for instance specifying for an answer that there is some uncertainty about the correctness of its content. Additive qualifiers enrich a communicative function with additional information, for instance adding that a request is accepted reluctantly. Of the qualifiers of this standard, "uncertain" and "conditional" are restrictive; sentiment qualifiers are additive.

The semantics of a communicative function qualified by a restrictive and by an additive qualifier is specified in (A.6) a. and b., respectively.

(A.6) a. $I_a(\langle f,q_r\rangle) = I_a(f)(I_a(q_r))$

b. $I_a(\langle f, q_a\rangle) = \lambda S.\ \lambda z.\ [I_a(f)(S,z) \cup I_a(q_a)(S,z)]$

This leads to the following interpretation of a dialogue act structure with qualifiers:

(A.7) $I_a(\langle S,A,d,f,q\rangle) = I_a(\langle f,q\rangle)(I_a(S), I_a(A), I_a(d))$

The following example of an uncertain *Inform* illustrates this.

(A.8) a. S: The KL 476 departs I think at 19:15.

b. $V_a(\text{Inform},\mathit{uncertain}) = [\lambda s.\ \lambda A.\ \lambda B.\ \lambda C_i.\ \lambda p.\ I_a(\text{Inform})(A,B,C_i,p,s)](I_a(\mathit{uncertain}))$

$= \lambda \text{A}.\ \lambda \text{B}.\ \lambda \text{C}_i.\ \lambda \text{p}.\ \text{U}_1(\text{A,B,C}_i,\text{p, weak}) \cup \text{U}_2(\text{A,B,C}_i,\text{p,weak})$

The update schemes U_1 and U_2 are defined as follows.

(A.9) $U_1(X,Y,D_i,p,s)$: add to component D_i of Y's pending context the information that participant X wants participant Y to believe that p with belief strength s.

$U_2(X,Y,D_i,p,s)$: add to component D_i of Y's pending context by extending the information that participant X believes that p with belief strength s.

A.4.3.2 Dependence relations

For a dialogue act without dependence relations the semantic content is determined by the functional segment that expresses the dialogue act, see (A.1). For a dialogue act that has a dependence relation to a set E of one or more entity structures, the semantic content is determined by the local semantic content of its functional segment together with that in the segments in E. Two combination functions are defined for combining the local semantic information and the information from the elements that the act depends on, one for functional dependences (κ_2) and one for feedback dependences (κ_3). This is expressed in (A.10):

(A.10) a. functional dependence: $I_{DA}(\langle m,\langle S,A,H,d,f,q,,E\rangle\rangle) = I_a(\langle S,A,H,d,f,q\rangle)(\kappa_2(\kappa_1(s), \{\kappa_1(s_i) \mid e_i \in E\}, f_i, f_a))$

b. feedback dependence: $I_{DA}(\langle m,\langle S,A,H,d,f,q,,E\rangle\rangle\rangle) = I_a(\langle S,A,H,d,f,q\rangle)(\kappa_3(s, \{e_i \mid e_i \in E\}, f_a))$

Note that the combinator κ_2 for functional dependences takes as its arguments not only the semantic interpretation of all the functional segments involved, but also the communicative function f_a of the responsive acts and the functions of its antecedents. For example, for an answer act responding to a propositional question, κ_2 delivers as value the semantic content of the question if the answer has the local content TRUE, as in the case of the answer "*Yes*", and it delivers the negation of that if the answer has local content FALSE.

A.4.3.3 Rhetorical relations

Link structures, annotating a rhetorical relation in a dialogue, can only be interpreted if a plug-in for DiAML is defined that specifies a set of relations (and argument roles). Their semantics then defined as an update operation which creates a relation between the dialogue acts concerned in the addressee's information state. This assumes that the dialogue acts that occur in a dialogue are explicitly represented in a part of an information state called the 'Dialogue History', an assumption that is shared by virtually all proposals for dialogue context modelling. The updates corresponding to link structures then come down to the addition of rhetorical relations between the dialogue act representations, with indication of argument roles and of semantic and pragmatic variations of these relations if that information is marked up in DiAML annotations.

Annex B
(normative)

DiAML-XML technical schema

This DiAML-XML representation relies on a three-level architecture:

a) a primary source, which may correspond to a speech recording, textual transcription, or any lower-level annotation thereof (e.g. a tokenization, or a morphosyntactic annotation);

b) the marking of functional segments from the primary source;

c) the actual dialogue act annotation associated with functional segments.

This annex provides a specification for this third level, as well as implementation guidelines for the two others.

The representation of a dialogue act annotated for a functional segment is done by means of the `dialogueAct` element. The attributes of this element and their values have been specified in Annex A. Functional relations between dialogue acts, like the relation between a question and an answer, or between an offer and its acceptance, are represented by the values of the `@functionalDependence` attribute; the relation between a dialogue act with a feedback function and the preceding dialogue act(s) that it provides or elicits feedback about, is likewise represented by the values of the `@feedbackDependence` attribute. Rhetorical relations among dialogue acts are represented by `rhetoricalLink` elements, which have an attribute `@rhetoRel` for specifying a particular rhetorical relation. The possible values of this attribute are not fixed by this standard, but would for example include such relations as `elaborate, justify, exemplify`.

Functional segments are identified by means of the `functionalSegment` element, which groups together the components of multimodal communicative behaviour that form a multimodal functional segment. The verbal component of a multimodal functional segment can be identified in terms of the words in a transcription of the spoken contribution, following TEI P5 for referring to the corresponding stretch of text using the `@span` attribute. The `spanGrp` element is available for grouping more than one contiguous span in order to represent a discontinuous stretch of speech. The `@target` attribute is used to point to a (possibly discontinuous) verbal segment, or to a nonverbal or multimodal stretch of dialogue behaviour.

The following example shows how the three levels, mentioned above, may be instantiated in the case of a tokenized primary source, encoded in accordance with the TEI guidelines. The source contains two utterances forming a small dialogue fragment, where the second utterance consists of a sentence interrupted by a filled pause ("... *um*..."), which is accompanied by a frowning expression and a head gesture, and followed by lip smacking and a sigh, before the verbal contribution continues.

(B.1) P1: Do you know where I should check in for Munich?

P2: For Munich go to ... um [+frown +waggle] [lip smack] [sigh] counters 31 to 40.

P2's utterance is segmented into two functional segments: the discontinuous verbal segment *"For Munich go to counters 31 to 40",* in which P2 expresses an answer to the preceding question, and the multimodal segment containing the frown, waggle, lip smack and sigh, plus the word "*um*"; in this segment P2 performs both a *Stalling* act and a *Turn Keep* act. Two alternative XML representations are shown of the dialogue act information associated with the primary data, one using the XML encoding of feature structures according to ISO 24610-1 and TEI P5, and compliant with W3C XML Schema in general; the other using a direct XML encoding of the DiAML concrete syntax, see 8.3.

The transcription of spoken or multimodal dialogue is not part of this standard, but the example shows how dialogue act annotations can be linked to XML representations of multimodal functional segments (see Reference [42] for further discussion of the issues involved). This example shows, for the sake of illustrating the possibilities, the XML representation of a multimodal segment that consists of a discontinuous verbal segment, a vocal component (heavily breathing out), a head movement (a 'waggle', i.e., left-right motion), a lip gesture (smacking), and an eyebrow gesture (frowning). Other components, like gaze direction or hand gestures, can be added in similar ways.

The TEI header contains metadata that include the identities of the dialogue participants.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-model href="http://www.tei-c.org/release/xml/tei/custom/schema/relaxng/tei_all.rng"
schematypens="http://relaxng.org/ns/structure/1.0"?>
<TEI xmlns="http://www.tei-c.org/ns/1.0">
  <teiHeader>
    <fileDesc>
      <titleStmt>
        <title>DiAML annotation example</title>
      </titleStmt>
      <publicationStmt>
        <p>...</p>
      </publicationStmt>
      <sourceDesc>
        <p>...</p>
      </sourceDesc>
    </fileDesc>
    <profileDesc>
      <particDesc>
        <person xml:id="p1">
          <p>the first participant</p>
        </person>
        <person xml:id="p2">
          <p>the second participant</p>
        </person>
      </particDesc>
    </profileDesc>
  </teiHeader>
  <text>
    <timeline unit="ms">
      <when xml:id="T1" absolute="192725"/>
      <when xml:id="T2" absolute="328377"/>
      <when xml:id="T3" absolute="357722"/>
      <when xml:id="T4" absolute="468737"/>
      <when xml:id="T5" absolute="488614"/>
      <when xml:id="T5" absolute="567512"/>
      <when xml:id="T6" absolute="715836"/>
      <when xml:id="T7" absolute="729126"/>
      <when xml:id="T8" absolute="761223"/>
      <when xml:id="T9" absolute="789264"/>
     <when xml:id="T10" absolute="881926"/>
     <when xml:id="T11" absolute="902804"/>
     <when xml:id="T12" absolute="1279207"/>
    </timeline>
    <body>
      <div>
        <head>Simple dialogue fragment</head>
         <u xml:id="u1" who="#p1" start="#T1" end="#T2">Do you know where I should
            check in for Munich</u>
         <u xml:id="u2a" who="#p2" start="#T3" end="#T4">For Munich go to</u>
         <u xml:id="u2b" who="#p2" start="#T5" end="#T6">um</u>
         <u xml:id="u2c" who="#p2" start="#T11" end="#T12">counters 31 to 40</u>
      </div>
      <div>
        <head>The dialogue turns, segmented into words (TEI- compliant)</head>
        <u>
          <w xml:id="w1">Do</w>
          <w xml:id="w2">you</w>
          <w xml:id="w3">know</w>
          <w xml:id="w4">where</w>
          <w xml:id="w5">I</w>
          <w xml:id="w6">should</w>
          <w xml:id="w7">check</w>
          <w xml:id="w8">in</w>
          <w xml:id="w9">for</w>
          <w xml:id="w10">Munich</w>
        </u>
        <u>
          <w xml:id="w11">For</w>
          <w xml:id="w12">Munich</w>
          <w xml:id="w13">go</w>
          <w xml:id="w14">to</w>
          <w xml:id="w15">um</w>
          <w xml:id="w16">counters</w>
          <w xml:id="w17">31</w>
          <w xml:id="w18">to</w>
          <w xml:id="w19">40</w>
        </u>
      </div>
      <div>
        <head>The nonverbal communicative behaviour of each of the participants, segmented
               and time-stamped)</head>
        <kinesic type="headMove" subtype="headGesture" xml:id="hmv1" who="#p2"
                start="#T5" end="#T6" ana="#gestDesc1#heg1"/>
        <kinesic type="browMove" subtype="frown" xml:id="bmv1" who="#p2" start="#T5"
                end="#T6"/>
        <kinesic type="lipMove" subtype="lipsmack" xml:id="lmv1" who="#p2" start="#T7"
                end="#T8"/>
        <vocal xml:id="voc1" who="#p2" type="outbreath" start="#T9" end="#T10"/>
        <kinesic type="headGesture" xml:id="heg1" ana="#gestDesc1"/>
    <fs xml:id="gestDesc1">
          <f name="direction">
            <symbol value="leftright"/>
          </f>
          <f name="velocity">
            <symbol value="slow"/></f>
    </fs>
      </div>
      <div>
        <head>Identification of functional segments</head>
        <fs type="verbalSegment" xml:id="ves1">
          <f name="segParts" fVal="#u1"/>
        </fs>
        <fs type="verbalSegment" xml:id="ves2">
          <f name="segParts" fVal="#u2a" "#u2c"/>
        </fs>
        <fsspanGrp type="functionalSegment" xml:id="fs1" ana="#da1">
          <f namespan type="verbalComponent" fVal="#ves1"xml:id="ts1" from="w1" to="w10"/>
        </fs>
        </spanGrp>
        <fsspanGrp type="functionalSegment" xml:id="fs2" ana="#da2">
          <f name="verbalComponent" fVal="#ves2"/>
        </fs>
        <fs type="functionalSegment" xml:id="fs3" ana="#da3 #da4" >
          <span type="verbalComponent" xml:id="ts2.1" from="w11" to="w19"/>
          <f namespan type="vocalComponent" fValfrom="#voc1"/>
          <f namespan type="headComponent" fValfrom="#hmv1"/>
          <f namespan type="lipComponent" fValfrom="#lmv1"/>
          <f namespan type="browComponent" fValfrom="#bmv1"/>
        </fs spanGrp>
      </div>
      <div>
        <head>Representation by means of feature structures in TEI/ISO- compliant format</
head>
        <fs type="dialogueAct" xml:id="da1" target="#fs1">
          <f name="sender" fVal="#p1"/>
          <f name="addressee" fVal="#p2"/>
          <f name="communicativeFunction">
            <symbol value="setQuestion"/></f>
          <f name="dimension">
            <symbol value="task"/></f>
          <f name="conditionality">
            <symbol value="conditional"/>
          </f>
        </fs>
        <fs type="dialogueAct" xml:id="da2" target="#fs2">
          <f name="sender" fVal="#p2"/>
          <f name="addressee" fVal="#p1"/>
          <f name="communicativeFunction">
            <symbol value="answer"/></f>
          <f name="dimension">
            <symbol value="task"/></f>
          <f name="functionalDependence" fVal="#da1"/>
        </fs>
        <fs type="dialogueAct" xml:id="da3" target="#fs3">
          <f name="sender" fVal="#p2"/>
          <f name="addressee" fVal="#p1"/>
          <f name="communicativeFunction">
               <symbol value="stalling"/></f>
        </fs>
        <fs type="dialogueAct" xml:id="da4" target="#fs3">
          <f name="sender" fVal="#p2"/>
          <f name="addressee" fVal="#p1"/>
          <f name="communicativeFunction">
               <symbol value="turnKeep"/></f>
        </fs>
      </div>
    </body>
  </text>
</TEI>
```

An alternative, direct XML encoding of DiAML annotation structures would look as follows, assuming the same representation of metadata and functional segments, but replacing the part from `<head>` Representation by means of feature structures in TEI/ISO-compliant format `</head>` until `</body>` by the XML lines below, enclosed within `<diaml ...>`, `</diaml>` brackets:

```
<diaml xmlns="https://www.iso.org/diaml">

<dialogueAct xml:id="da1" target="#fs1" sender="#p1" addressee="#p2" dimension="task"
    communicativeFunction="setQuestion" conditionality="conditional"/>
    <dialogueAct xml:id="da2" target="#fs2" sender="#p2" addressee="#p1" dimension="task"
    communicativeFunction="answer" functionalDependence="#da1"/>
</diaml>
```

Annex C
(normative)

Data categories for DiAML concepts

C.1 Overview

This annex contains data categories for dimensions, communicative functions, and qualifiers. A data category, as defined in ISO 12620, has the definition of a concept as its most important part. A definition has a *Source* attribute, which indicates the origin of the definition, and a *Note* attribute that may be used e.g. for mentioning alternative and related terms and concepts.

Two optional components of a data category specification are a *Conceptual domain*, which lists the special cases of the defined concept, and a *Broader concept,* which can be used to indicate that a concept is a special case of a more general concept. For example, the **/answer/** data category has the conceptual domain **/confirm/**, **/disconfirm/**, and the broader concept **/inform/**. Together, the values of these two components can be used to define a hierarchical structure in a set of concepts, such as the hierarchy of general-purpose communicative functions shown in Figure 2.

C.2 Dimensions

/task/	
Definition	Category of dialogue acts whose performance contributes to pursuing the task or activity that motivates the dialogue.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Task and Task Management (DAMSL), Activity (GBG-IM), Task/Activity (DIT).
Explanation	The notion of a 'task' is intended in a very broad sense here, including any activity which can be said to aim at achieving a goal. Such a goal may be quite specific, such as knowing the arrival time of a particular train, or more general, such as creating a pleasant atmosphere.

/autoFeedback/	
Definition	Category of dialogue acts by which the sender discusses or reports on his processing of previous dialogue contributions.
-- Source	Bunt, 1995
-- Note	Related terminology in other schemes: Feedback (GBG-IM); Backchannel (common). Feedback in GBG-IM includes the class feedback elicitation acts which forms part of the /alloFeedback/ category.

/alloFeedback/	
Definition	Category of dialogue acts in which the sender discusses the addressee's processing of previous dialogue contributions.
-- Source	Bunt, 1995

/turnManagement/	
Definition	Category of dialogue acts whose performance is intended to regulate the allocation of the speaker role.
-- Source	Allwood et al., 1993
-- Note	In the literature often referred to as the "turn-taking system".

/timeManagement/	
Definition	Category of dialogue acts which concern the allocation of time to the participant occupying the speaker role.
-- Source	DIT

/discourseStructuring/	
Definition	Category of dialogue acts which explicitly structure the interaction.
-- Source	DIT

/ownCommunicationManagement/	
Definition	Category of dialogue acts by which the speaker edits his own speech within the current turn.
-- Source	Allwood et al., 1993

/partnerCommunicationManagement/	
Definition	Category of dialogue acts which are performed by a dialogue participant who does not have the speaker role and by which he/she edits the speech of the participant who currently has the speaker role.
-- Source	DIT++

/socialObligationsManagement/	
Definition	Category of dialogue acts performed for dealing with social obligations such as greeting, thanking, and apologizing.
-- Source	DIT

'

/contactManagement/	
Definition	Category of dialogue acts which are performed by a dialogue participant for establishing or ensuring contact with other participants.
-- Source	DIT++

C.3 Communicative functions

C.3.1 General-purpose functions

C.3.1.1 Information-seeking functions

/question/	
Conceptual domain	/setQuestion/ /propositionalQuestion/ /choiceQuestion/ /checkQuestion/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to obtain the information, described by the semantic content, which S assumes that the addressee, A, possesses; S puts pressure on A to provide this information.
-- Source	Commonplace
-- Note	The notion of 'question' defined only covers those cases where the sender genuinely wants to obtain the information that he/she is asking about. It does not include for instance 'exam questions', where the speaker does know the answer to his question, but wants to know whether the examinee also knows it, not does it include rhetorical questions, which from a semantic point of view are not questions at all but rather the expression of an opinion.
Example	"And so?"
-- Source	DIAMOND corpus

/propositionalQuestion/	
Conceptual domain	/checkQuestion/
Broader concept	/question/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to know whether the proposition, described by the semantic content, is true. S assumes that the addressee, A, knows whether the proposition is true, and puts pressure on A to provide this information.
-- Source	LIRICS
-- Note	Related terminology in other schemes: YN-Question (TRAINS), Query-yn (HCRC Map Task); info-request (DAMSL).
Explanation	A propositional question corresponds to what is commonly termed a YN-question in the linguistic literature. This standard prefers the term 'propositional question' because the term 'YN-Question' carries the suggestion that this kind of question can only be answered by "yes" or "no", which is actually not the case.
Example	"Does the meeting start at ten?"

/setQuestion/	
Broader concept	/question/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to know which elements of a given set have a certain property specified by the semantic content. S puts pressure on the addressee, A, to provide this information, which S assumes that A possesses. S believes that at least one element of the set has that property.
-- Source	LIRICS
-- Note	Related terminology in other schemes: WH-Question (SWBD-DAMSL, MRDA), Query-w (HCRC MapTask), and WHQ (TRAINS).
Explanation	A set question corresponds to what is commonly termed a WH-question in the linguistic literature. The term 'set question' is preferred because: (a) it clearly separates form from function by removing any oblique reference to syntactic criteria for the identification of such acts; and (b) it is not a language specific term (it may be further noted that even in English, not all questioning words begin with 'wh', e.g. "How?").
Example	"What time does the meeting start?"; "How far is it to the station?"

/checkQuestion/	
Broader concept	/propositionalQuestion/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to know whether a proposition, which forms the semantic content, is true. S holds the uncertain belief that it is true. S assumes that A knows whether the proposition is true or not, and puts pressure on A to provide this information.
-- Source	LIRICS
-- Note	Related terminology in other schemes: Check (DIT, HCRC MapTask, TRAINS), Tag Question (SWBD-DAMSL), Request_Comment (Verbmobil).
Example	"The meeting starts at ten, right?"

/choiceQuestion/	
Broader concept	/question/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to know which one from a list of alternative propositions, specified by the semantic content, is true; S believes that exactly one element of that list is true; S assumes that the addressee, A, knows which of the alternative propositions is true, and S puts pressure on A to provide this information.
-- Source	DIT++
-- Note	Related terminology in other schemes: Alternatives Question (DIT, LIRICS), QUERY-W (HCRC MapTask), Or-Question/Or-Clause (SWBD-DAMSL, MRDA). Also commonly known as 'menu question' or 'multiple-choice question'.
Example	"Should the telephone cable go in the telephone line slot or in the external line slot?"
-- Source	DIAMOND corpus

/testQuestion/	
Broader concept	/question/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to know whether the addressee, A, possesses the requested information, which S does possess. S puts pressure on A to provide the requested information.
-- Source	Commonplace
-- Note	Test questions have the linguistic same form as ordinary questions; their occurrence in a particular setting with participants in specific roles (such as tutor and student) makes a question recognisable as a test question.

C.3.1.2 Information-providing functions

/inform/	
Conceptual domain	/agreement/ /disagreement/ /answer/ /confirm/ /disconfirm/ /correction/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make the information contained in the semantic content available to the addressee, A; S assumes that the information is correct.
-- Source	DIT++
-- Note	Related terminology in other schemes: Assert (DAMSL, COCONUT), Statement (SWBD-DAMSL, MRDA, MALTUS).
Explanation	The inform function may also have more specific rhetorical functions such as: explain, elaborate, exemplify and justify; this is treated in this standard by means of rhetorical relations.
Example	"The 6.34 to Breda leaves from platform 2."

/agreement/	
Broader concept	/inform/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S assumes a given proposition to be true, which S believes that A also assumes to be true.
-- Source	DIT++
-- Note	Related terminology in other schemes: Accept (DAMSL, SWBD-DAMSL, TRAINS, Verbmobil, MALTUS, SPAAC).
Explanation	DAMSL and SWBD-DAMSL use “Agreement" to refer to various degrees in which some previous proposal, plan, opinion or statement is accepted; "accept" is one of these degrees; “reject" is another.
Example	English: "Exactly"; Dutch: "Precies!"; Danish: "Netop!"

/disagreement/	
Conceptual domain	/correction/
Broader concept	/inform/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S assumes a given proposition to be false, which S believes that A assumes to be true.
-- Source	DIT++
-- Note	Related terminology in other schemes: Reject (DAMSL, COCONUT, TRAINS, MRDA, Verbmobil); Denial (TRAINS).
Explanation	DAMSL and SWBD-DAMSL use “Agreement" to refer to various degrees in which some previous proposal, plan, opinion or statement is accepted; "accept" is one of these degrees; “reject" is another.
Example	J: "do you know where to find ink savings?" S: "uh... oh I think to the left of the ink cartridge" J: "uh... no"
-- Source	DIAMOND corpus

/correction/	
Broader concept	/disagreement/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that certain information which S has reason to believe that A assumes to be correct, is in fact incorrect and that instead the information that S provides is correct.
-- Source	Commonplace
Example	"To Montreal, not to Ottawa"

/answer/	
Conceptual domain	/confirm/ /disconfirm/
Broader concept	/Inform/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make certain information available to the addressee, A, which S believes A wants to know; S assumes that this information is correct.
-- Source	Commonplace
Example	S: "what does the display say?" H: "send error document ready"
-- Source	DIAMOND corpus

/confirm/	
Broader concept	/answer/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that the proposition which forms the semantic content is true. S believes that A holds a weak belief that this proposition is true, and that A wants to know for certain whether it is; S assumes that it is.
-- Source	Commonplace
Example	"Indeed"

/disconfirm/	
Broader concept	/answer/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that the proposition which forms the semantic content is false. S believes that A holds a weak belief that this proposition is true, and that S wants to know for certain whether it is; S assumes that it is false.
-- Source	DIT++
-- Note	Related terminology in other schemes: Reply-N (HCRC MapTask); No-Answer (SWBD-DAMSL); Dispreferred answer (MRDA).
Example	French "si"; Danish "jo"; Dutch: "toch niet" and "toch wel"; German: "doch"

C.3.1.3 Commissive functions

/offer/	
Conceptual domain	/promise/
Definition	Communicative function of a dialogue act by which the sender, S, commits himself/herself to perform the action, specified by the semantic content, in the manner or with the frequency that may be specified, conditional on the consent of the addressee that S do so.
-- Source	Commonplace
-- Note	Related terminology in other schemes:
Example	"I will look that up for you"

/promise/	
Broader concept	/offer/
Definition	Communicative function of a dialogue act by which the sender, S, commits himself/herself to perform the action, specified in the semantic content, in the manner or with the frequency that may be specified. S believes that this action would be in the interest of the addressee.
-- Source	Searle (1969)
-- Note	Related terminology in other schemes: Commit (DAMSL, COCONUT, Verbmobil, MALTUS); Commitment (MRDA); Inform Intent (SPAAC)
Example	"Shall I begin?"; "Would you like to have some coffee?"

/addressRequest/	
Conceptual domain	/acceptRequest/ /declineRequest/
Definition	Communicative function of a dialogue act by which the sender, S, indicates that he/she considers performing an action that he/she was requested to perform.
-- Source	DIT++
-- Note	Related terminology in other schemes: Assess (AMI).
Explanation	The addressRequest function covers a range of possible responses to a request. If the response does not contain the expression of a condition, then the sender commits himself/herself unconditionally to perform the requested action; this is the special case of /acceptRequest/. If the condition is specified that the action be performed zero times, then the sender in fact declines to perform the requested action (as he/she commits himself/herself to *not* perform the action).
Example	A: "Give me the gun."
	S: "If you push the bag to me"

/acceptRequest/	
Broader concept	/addressRequest/
Definition	Communicative function of a dialogue act by which the sender, S, commits himself/herself to performing an action that he/she was requested to perform.
-- Source	LIRICS
-- Note	Related terminology in other schemes: Accept (DAMSL, SWBD-DAMSL, TRAINS, Verbmobil).
Example	A: "Could you close the door please?" B: "Sure."

/declineRequest/	
Broader concept	/addressRequest/
Definition	Communicative function of a dialogue act by which the sender, S, commits himself/herself to not perform an action that he/she was requested to perform.
-- Source	LIRICS
-- Note	Related terminology in other schemes: Reject (DAMSL, SWBD-DAMSL, TRAINS, Verbmobil).
Example	"Not now."

/addressSuggest/	
Conceptual domain	/acceptSuggest/ /declineSuggest/
Definition	Communicative function of a dialogue act by which the sender, S, indicates that he/she considers to perform an action that was suggested to him/her, possibly depending on certain conditions that he/she makes explicit.
-- Source	DIT++
-- Note	Related terminology in other schemes: Assess (AMI).
Example	A: "Let's go together."
	S: "Only if we're in full agreement about how to proceed when we get there."

/acceptSuggest/	
Broader concept	/addressSuggest/
Definition	Communicative function of a dialogue act by which the sender, S, commits himself/herself to perform an action that was suggested to him/her, possibly with certain restrictions or conditions concerning manner or frequency of performance.
-- Source	LIRICS
-- Note	Related terminology in other schemes: Accept (DAMSL, SWBD-DAMSL, TRAINS, Verbmobil).
Example	A: "Shall we go and have a look around?" B: "Let's do so."

/declineSuggest/	
Broader concept	/addressSuggest/
Definition	Communicative function of a dialogue act performed by which the sender, S, indicates that he/she will not perform an action that was suggested to him/her, possibly depending on certain conditions that he/she makes explicit.
-- Source	LIRICS
-- Note	Related terminology in other schemes: Reject (DAMSL, SWBD-DAMSL, TRAINS, Verbmobil).
Example	"I'd rather not."

C.3.1.4 Directive functions

/request/	
Conceptual domain	/instruct/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make the addressee, A, feel obliged to perform a certain action in the manner or with the frequency described by the semantic content, conditional on A's consent to perform the action. S assumes that A is able to perform this action.
-- Source	DIT++
-- Note	Related terminology in other schemes: Influence-addressee-future-action (DAMSL); Request Commit (Verbmobil).
Example	"Please turn to page five"; "Don't do this ever again, please".

/instruct/	
Broader concept	/request/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make the addressee, A, feel obliged to perform a certain action which is described in or can be inferred from the semantic content, in the manner or with the frequency described by the semantic content. S assumes that A is able to perform this action.
-- Source	DIT++; HCRC Map Task
-- Note	Related terminology in other schemes: Action-directive (DAMSL, SWBD-DAMSL, COCONUT); Command (HCRC Map Task); Direct (SPAAC); Do (MALTUS)
Example	"Go right round until you get to just above that."
-- Source	HCRC Map Task corpus

/suggest/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make the addressee, A, consider the performance of a certain action, specified by the semantic content. S believes that this action is in A's interest, and assumes that A is able to perform the action.
-- Source	DIT++
-- Note	Related terminology in other schemes: Open-option (DAMSL, SWBD-DAMSL, COCONUT).
Example	"Let's wait for the speaker to finish."

/addressOffer/	
Broader concept	/instruct/
Conceptual domain	/acceptOffer/ /declineOffer/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to indicate that he/she is considering the possibility that A performs the action that A has offered to perform, possibly with certain conditions that he/she makes explicit.
-- Source	DIT++
-- Note	Related terminology in other schemes: Assess (AMI).
Example	"Yes please"; French: "Je vous en prie"

/acceptOffer/	
Broader concept	/addressOffer/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S would like A to perform the action that A has offered to perform, possibly with certain conditions that he/she makes explicit.
-- Source	LIRICS
-- Note	Related terminology in other schemes: Accept (DAMSL, SWBD-DAMSL, TRAINS, Verbmobil).
Example	"Yes please"; French: "Je vous en prie"; Dutch: "Graag"; German: "Bitte"

/declineOffer/	
Broader concept	/addressOffer/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S does not want A to perform the action that A has offered to perform, possibly depending on certain conditions that he/she makes explicit.
-- Source	LIRICS
-- Note	Related terminology in other schemes: Reject (DAMSL, SWBD-DAMSL, TRAINS, Verbmobil).
Example	English: "No thank you"; Danish: "Nej tak"; French: "Non merci".

C.3.2 Dimension-specific functions

C.3.2.1 Feedback functions

/autoPositive/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S believes that S's processing of the previous utterance(s) was successful.
-- Source	LIRICS
-- Note	Related terminology in other schemes: Signal-Understanding (DAMSL, MRDA), Acknowledgement (HCRC MapTask, TRAINS, SPAAC, C-Star), Backchannel (Verbmobil). Feedback-Positive (Verbmobil). This type of feedback may be further broken down into specific levels of processing (dealing with the sender's attention, perception, interpretation, evaluation and execution), as exemplified in the DIT and SLSA schemes.
Explanation	Feedback mostly concerns the processing of the last utterance from the addressee, but sometimes, especially in the case of positive feedback, it concerns a longer stretch of dialogue.
Example	"Uh-huh"; "Okay"; Nonverbally: nodding; "Yes"

/autoNegative/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A that S's processing of the previous utterance(s) encountered a problem.
-- Source	LIRICS
-- Note	Related terminology in other schemes: Signal-Non-Understanding (DAMSL, Coconut, MRDA), Pardon (SPAAC), Feedback-Negative (Verbmobil). This type of feedback may be further broken down into more specific levels of processing, as is exemplified in the DITand SLSA schemes.
Example	English: "I beg you pardon"; Portuguese: "Como?"

/alloPositive/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S believes that A's processing of the previous utterance(s) was successful.
-- Source	LIRICS
-- Note	This type of feedback may be further broken down into more specific levels of processing, as in the DIT^{++} and SLSA schemes.
Example	"Correct"; "Right"

/alloNegative/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S believes that A's processing of the previous utterance(s) encountered a problem.
-- Source	LIRICS
-- Note	This type of feedback may be broken down into more specific levels of processing, as is done in the DIT^{++} scheme.
Example	"No no no no no"

/feedbackElicitation/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to know whether A's processing of the previous utterance(s) was successful.
-- Source	Allwood et al., 1993
-- Note	Related terminology in other schemes: Request Clarify (Verbmobil), Understanding Check (MRDA), Clarification Check (COCONUT), Check (HCRC Map Task), Question Attention (MALTUS).
Example	English: "Okay?"; Italian: "Capisce?"; Dutch: "Ja?"

C.3.2.2 Turn-management functions

/turnAccept/	
Broader concept	/turnTake/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to signal his willingness to take the speaker role, as requested by the previous speaker.
-- Source	Common in literature on turn taking in conversation.
-- Note	Occurs especially in multiparty dialogue. Related terminology in other schemes: Take-Turn (TRAINS), Turn Opening (SLSA).
Example	A: "What do you say, Craig?" C: "OK, let me see."
-- Source	AMI corpus

/turnTake/	
Conceptual domain	/turnAccept/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to have the speaker role, which is available at that moment.
-- Source	Common in literature on turn taking in conversation.
-- Note	Related terminology in other schemes: Turn-Take (TRAINS), Regain Turn (MRDA).
Example	"Uh..." as a turn-initial segment
-- Source	

/turnGrab/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to take the speaker role away from the participant who currently occupies it.
-- Source	Common in literature on turn taking in conversation.
-- Note	Related terminology in other schemes: Grabber (MRDA); Turn Grabber (MALTUS, Primula); Interruption (SLSA).
Example	“Hold on”; nonverbally: sticking up a hand as a stop signal

/turnAssign/	
Broader concept	/turnRelease/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to pass the speaker role to a designated other participant.
-- Source	Common in literature on turn taking in conversation.
-- Note	Related terminology in other schemes: Turn Give (DIT), Assign-Turn (TRAINS).
Example	A: "Craig?", characteristically accompanied by the speaker directing his gaze to Craig, possibly also nodding or pointing in his direction and raising the eyebrows.
-- Source	AMI corpus

/turnRelease/	
Conceptual domain	/turnAssign/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to give other dialogue participants the opportunity to occupy the speaker role.
-- Source	Common in literature on turn taking in conversation.
-- Note	Related terminology in other schemes: Turn closing (SLSA).
Example	Sender uses declining intonation towards the end of a contribution and subsequently pauses.

/turnKeep/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to keep the speaker role.
-- Source	Common in literature on turn taking in conversation.
-- Note	Related terminology in other schemes: Turn maintain (DAMSL, SWBD-DMSL); Holder (MRDA); Hold (SPAAC, Chiba); Turn holder (MALTUS, Primula); Turn holding (SLSA). Note: utterances used for turn keeping often also have a stalling function.
Example	"Uh" not in turn-initial position

C.3.2.3 Time-management functions

/stalling/	
Definition	Communicative function of a dialogue act performed in order to have a little more time for constructing his contribution.
-- Source	DIT++
-- Note	Related terminology in other schemes: Hold (SPAAC); Stall (AMI); Delay (DAMSL, SWBD-DAMSL, COCONUT). Turn-initial segments with a Stalling function often also have a Turn Take or Turn Accept function; segments inside a turn which have a Stalling function often also have a Turn Keep function.
Example	"Let me see...", "Uh..."; speaking slowly; repeating something ("We... we went to...")

/pausing/	
Definition	Communicative function of a dialogue act performed in order to suspend the dialogue for a short while.
-- Source	DIT++
-- Note	Related terminology in other schemes: Pause (Alparon); Please wait (C-Star); Hold before answers (MRDA).
Explanation	Pausing occurs either in order to prepare a continuation of the dialogue (e.g. the sender needs to look up something), or because something else came up which is more urgent for the sender to attend to.
Example	English: "Just a moment"; Danish: "Lige et øjeblik"; Dutch: "Een ogenblikje"

C.3.2.4 Discourse-structuring functions

/interactionStructuring/	
Conceptual domain	/opening/
Definition	Communicative function of a dialogue act performed in order to explicitly structure the interaction.
-- Source	LIRICS
-- Note	The function “Interaction structuring" covers a range of activities which explicitly structure the dialogue, such as the introduction of a new topic, the announcement of a certain type of dialogue act, and the closing of a topic.
Example	"And the windows, we had to replace all the windows"
-- Source	Switchboard corpus

/opening/	
Broader concept	/interactionStructuring/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S is ready and willing to engage in a dialogue with A.
-- Source	DAMSL
-- Note	Related terminology in other schemes: Task Initiate (Verbmobil).
Example	"Okay" at the start of a (multi-party) dialogue
-- Source	AMI corpus

/topicShift/	
Broader concept	/interactionStructuring/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S is going to continue the dialogue on a different topic.
-- Source	DIT
Example	"Something else."

C.3.2.5 Own- and partner-management functions

/selfError/	
Conceptual domain	/selfCorrection/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to signal to the addressee, A, that S has made a mistake in speaking.
-- Source	DIT++
-- Note	Related terminology in other schemes: Repaired (TRAINS); Change (SLSA).
Example	S: "so you want to leave at eight o'clock in the morning?"
	U: "yes oh sorry no..."
-- Source	OVIS corpus

/retraction/	
Conceptual domain	/selfCorrection/
Broader concept	/selfError/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to withdraw something he/she just said within the same turn.
-- Source	DIT++
-- Note	Related terminology in other schemes: Speech Repair (DAMSL, MRDA, TRAINS), Repair (TRAINS), Correct-Self (SPAAC).
Example	"then we're going to g-- "
-- Source	HCRC Map Task corpus

/selfCorrection/	
Broader concept	/retraction/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to correct a speaking error that he/she just made, or to improve on an infelicitous formulation that he/she just used, within the same turn.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Speech Repair (DAMSL, MRDA, TRAINS); Correct-self (SPAAC); Correct-Assumption (COCONUT).
Example	"then we're going to g-- ... turn straight back"
-- Source	HCRC Map Task corpus

/completion/	
Definition	Communicative function of a dialogue act performed for assisting the addressee in the completion of an utterance.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Complete (SPAAC); Collaborative completion (MRDA).
Example	A: "which should leave us plenty of time to uh... uh" S: "get to Corning"
-- Source	TRAINS corpus

/correctMisspeaking/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to correct (part of) an utterance by the addressee, A, assuming that A made a speaking error.
-- Source	DAMSL
-- Note	Related terminology in other schemes: Correct Misspeaking (DIT); Correction suggestion (TRAINS).
Example	A: "second engine E3 is going to uh Corning to pick up the bananas, back to Avon, drop..." S: "to pick up the oranges" A: "sorry, pick up the oranges"
-- Source	TRAINS corpus

C.3.2.6 Social obligations management functions

/initGreeting/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S is present and aware of A's presence; S puts pressure on A to acknowledge this.
-- Source	DIT++
-- Note	Related terminology in other schemes: Greeting (DAMSL, SWBD-DAMSL, COCONUT, C-Star), Greet (Verbmobil, SLSA, TRAINS, Alparon).
Explanation	Greetings usually come in initiative-response pairs within a dialogue; this data category corresponds to the first element of such a pair.
Example	"Hello!"; "Good morning"; "How are you?"

/returnGreeting/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to acknowledge that S is aware of the presence of the addressee, A, and of A having signalled his presence to S.
-- Source	DIT++
-- Note	Related terminology in other schemes: Greeting (DAMSL, SWBD-DAMSL, COCONUT, C-Star), Greet (Verbmobil, SLSA, TRAINS, Alparon).
Explanation	Greetings usually come in initiative-response pairs within a dialogue; this data category corresponds to the second element of such a pair.
Example	I: "Schiphol Information, good morning."
	C: "Good morning".

/initSelfIntroduction/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make himself/herself known to the addressee, A; S puts pressure on A to acknowledge this.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Introduce (Verbmobil).
-- Explanation	Introductions usually come in initiative-response pairs within a dialogue; this data category corresponds to the first element of such a pair.
Example	"Schiphol Information."

/returnSelfIntroduction/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make himself/herself known to the addressee, A, in response to a self-introduction by A.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Introduce (Verbmobil).
-- Explanation	Introductions usually come in initiative-response pairs within a dialogue; this data category corresponds to the second element of such a pair.
Example	I: "Schiphol Information, good morning."
	C: "Good morning, this is De Bruin in Arnhem."

/apology/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to signal that he/she wants the addressee, A, to know that S regrets something; S puts pressure on A to acknowledge this.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Apologise (C-Star); Polite (Verbmobil).
Example	A: "second engine E3 is going to uh Corning to pick up the bananas, back to Avon, drop..."
	S: "to pick up the oranges"
	A: "sorry, pick up the oranges"
-- Source	TRAINS corpus

/acceptApology/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to mitigate, the feelings of regret that the addressee, A, has expressed.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Downplayer (SWBD-DAMSL, MRDA)
Example	"No problem."

/thanking/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to inform the addressee, A, that S is grateful for some action performed by A; S puts pressure on A to acknowledge this.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Thank (Verbmobil).
Example	English: "Thanks a lot."; Portuguese: "Muito obrigado"; Swedish: "Tack so mycket", Greek: "Evcharisto"

/acceptThanking/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to mitigate to the feelings of gratitude which the addressee, A, has expressed.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Downplayer (SWBD-DAMSL).
Example	English: "Don't mention it"; Spanish: "De nada"; Greek: "parakalo".

/initGoodbye/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to signal the current utterance is his last contribution to the dialogue; S pressures the addressee, A, to respond with a returnGoodbye act.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Goodbye (DAMSL, COCONUT), Bye (Verbmobil).
Example	S: "Bye bye, see you later"

/returnGoodbye/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to acknowledge his awareness that the addressee, A, has made his last contribution to the dialogue and to signal his agreement to end the dialogue.
-- Source	Commonplace
-- Note	Related terminology in other schemes: Bye (Verbmobil).
Example	(S: "Bye bye, see you later") A: "Bye bye, see you."

/compliment/	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make the addressee, A, aware that S likes A's appearance, certain of A's qualities, or something that A has achieved.
-- Source	Commonplace
Explanation	Follow-on greetings usually come in response to opening greetings and/or self-introductions.
Example	"Well done!", "You look great".

/congratulation /	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make the addressee, A, aware that the speaker takes pleasure in the successful achievement of something by A or in A's good fortune.
-- Source	After Merriam-Webster.
Example	"Congratulations!".

/sympathyExpression /	
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make the addressee, A, aware that the speaker is sorry for something that happened to the addressee.
-- Source	
Example	"I'm sorry to hear that?".

C.3.2.7 Contact management functions

/contactCheck/	
Broader concept	/question/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to verify that the addressee, A, is ready to communicate with S.
-- Source	DIT++
-- Example	"Yes?"; "Hello?"

/contactIndication/	
Broader concept	/question/
Definition	Communicative function of a dialogue act performed by the sender, S, in order to make it known to the addressee, A, that S is ready to communicate with A.
-- Source	DIT++
-- Example	"Yes"; "Oh hi"

C.4 Qualifiers

C.4.1 Conditionality

/conditionality/	
Definition	Class of predicates which can be associated with most action-discussion functions to express whether the sender of a dialogue act with that function is considering the performance of the action under discussion subject to certain conditions.
-- Source	Petukhova and Bunt (2009a)

/conditional/	
Definition	Predicate which can be associated with most action-discussion functions to express that the sender of a dialogue act with that function is considering the performance of the action under discussion subject to certain conditions.
-- Source	Petukhova and Bunt (2009a)
Example	"If you're ready maybe you can start the presentation"
-- Source	AMI corpus
Example	A: "Can we just go over that again"
	B: "I'm afraid we don't have time, unless you do it very quickly"

/unconditional/	
Definition	Predicate which can be associated with an action-discussion function to express that the sender of a dialogue act with that function is considering the performance of the action under discussion without any conditions.
-- Source	Petukhova and Bunt (2009a)
Example	"I'll come tomorrow, no matter what."

C.4.2 Certainty

/certainty/	
Definition	Class of predicates which can be associated with a communicative function to express whether the sender of a dialogue act with that function is certain or uncertain about the correctness of the information that he/she provides.
-- Source	DIT++

/uncertain/	
Definition	Predicate which can be associated with a communicative function to express that the sender of a dialogue act with that function is uncertain about the correctness of the information that he/she provides.
-- Source	Petukhova and Bunt (2009a)
Example	"That might be a good idea."
-- Source	AMI corpus

/certain/	
Definition	Predicate which can be associated with a communicative function to express that the sender of a dialogue act with that function is certain about the correctness of the information that he/she provides.
-- Source	Petukhova and Bunt (2009a)

C.4.3 Sentiment

/sentiment/	
Definition	Class of predicates which can be associated with a communicative function to express an attitude of the sender towards the semantic content of the dialogue act.
-- Source	Petukhova and Bunt (2009a)
-- Note	In the absence of a widely agreed set of sentiment values, this standard does not define any data categories for sentiment values.

/positive/	
Definition	Predicate which expresses that the sender of a dialogue act with that function feels good about the semantic content of the dialogue act.
-- Source	
Example	

/negative/	
Definition	Predicate which expresses that the sender of a dialogue act with that function does not feel good about the semantic content of the dialogue act.
-- Source	
Example	

Annex D
(informative)

Plug-ins for semantic content and other enrichments

D.1 Overview

This annex describes the use of tripartite (or "triple-layered") annotation plug-ins introduced in 9.3. Clause D.2 describes three alternative plug-ins for enriching dialogue act annotations with information about their semantic content. Clauses D.3 to D.5 describe other plug-ins that can be used to enrich dialogue act annotations or to make them more accurate, more specifically for adding the possibility of marking up rhetorical relations in dialogue by importing concepts from ISO 24617-8 (dealing with discourse relation annotation); for adding communicative functions for more fine-grained annotation of feedback and casual talk; for adding application-specific communicative functions; and for adding the possibility of marking up emotional aspects of the performance of a dialogue act.

D.2 Plug-ins for semantic content

For the use cases UC3 and UC4 mentioned in Clause 4, it is crucially 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 quantification and modification phenomena.

Following ISO 24617-6, the specification of a semantic annotation scheme is a structured triple (see 9.3).

(D.1) $L_a = \langle AS_a, CS_a, Sm_a \rangle = \langle \langle CI_a, AC_a \rangle, \langle V_a, CC_a, F_a \rangle, \langle M_a, I_a \rangle \rangle$

An ***annotation plug-in*** is an annotation scheme that can be added on to a host annotation scheme, in general defining additions to all three parts of the host scheme (AS_a, CS_a, and Sm_a in (D.1)) and having the same tripartite structure, introducing (1) additional entity structures and link structures in the abstract syntax, (2) the semantics of these structures, and (3) their encodings. To combine elements from the host and plug-in schemes, a ***plug-in interface*** is needed which defines link structures that relate entity structures of the two schemes. A plug-in for semantic content introduces for example a <contentLink> element for linking a dialogue act annotation in DiAML-XML with a semantic content structure in XML [see (D.5)]. A plug-in has again the same three-part structure as the host and plug-in schemes, but some of its parts are empty since it only introduces structures for 'linking' the structures of the host and the plug-in scheme. This is shown in (D.2) for the interface ${}_aY_c$ of host annotation scheme L_a and content plug-in PL_c.The interface semantics specifies the meaning of the content link structure 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 formed by the semantics of the content structure 'c', i.e. $I_c(c)$, as defined by the plug-in semantics. This reflects the dialogue act theory underlying this document, according to which the semantics of a full-blown dialogue act is an update operation on information states, obtained by applying the semantics of the functional characterization of the dialogue act to its semantic content [see (A.2) in Annex A].

(D.2) $_aY_c = \langle _aAS_c, {}_aCS_c, {}_aSm_c\rangle$, with:

$_aAS_c = \langle {}_aCI_c, {}_aAC_c\rangle = \langle Ø, \{\langle a,c\rangle \text{ content link structure}\}\rangle$

$_aCS_c = \langle {}_aV_c, {}_aCC_c\rangle, {}_aF_c\rangle \rangle = \langle\langle Ø, \{\text{<contentLink> element}\}, {}_aF_c\rangle(\langle a, c\rangle) =$

<contentLink dialAct=$_aF_c$(a) semContent= $_aF_c$(c)/>⟩

$_aSm_c = \langle {}_aM_c, {}_aI_c\rangle = \langle\langle Ø, {}_aI_c(\langle a,c\rangle) = I_a(a)(I_c(c))\rangle$

Host annotation scheme, plug-in and interface together define an extended annotation scheme formed by the unions of their components. This is a useful extended dialogue act annotation scheme only if two important properties of the host annotation scheme are preserved: the orthogonality of the dimensions and the taxonomical structure of the set of communicative functions.

D.2.1 Attribute-Value plug-in

A simple domain-specific plug-in for semantic content described as (lists of) attribute-value pairs can for example be useful in a travel domain where a journey can be described by a few attribute-value pairs, specifying departure place, destination, travel date, etc. In such a context, when a client P1 says *"I'd like to leave around ten in the morning"*, this can be annotated as in (D.3b).

(D.3) a. P1: I'd like to leave around ten in the morning (= markable m1)

b. `<avContent xml:id="c1" target="#m1" attribute="departureTime" value="10:00"/>`

Underlying the representation in (D.3b) 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 a feature structure $[A_i': v_{ij}']$ which, according to ISO 24612 for feature structures, expresses the property $\lambda x. A_i'(x) = v_{ij}'$. The variable 'x' in the lambda abstraction can in this domain be thought of as ranging over journeys. The syntax and semantics of such AV-entity structures define a very simple annotation language L_{AV}, the semantics of which is defined in (E.4).

(D.4) $I_{AV}(\langle A_i, v_{ij}\rangle) = [I_{AV}\}(A_i): I_{AV}\}(v_{ij})] = [A_i': v_{ij}']$

To link an AV-content annotation to dialogue act annotations, the XML element `<contentLink>`, defined in the interface (D.2), can be used to obtain representations of the form (D.5).

(D.5)
```
<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"/>
```

The formal specification of the attribute-value content plug-in PL_{AV} for DiAML annotations is as follows:

— **Abstract syntax:** $AS_{AV} = \langle CI_{AV}, AC_{AV}\rangle$:

— the conceptual inventory CI_{AV} lists attributes and their possible values;

— AC_{AV}: content entity structures are of the form $\langle m, \langle A_i, v_{ij}\rangle\rangle$[5], with m a markable;

— **Concrete syntax**: $CS_{AV} = \langle V_{AV}, CC_{AV}, F_{AV}\rangle$:

— the vocabulary VC_{AV} lists names of XML attributes and values;

— CC_{AV}: specification of XML element <avContent>, see (D.3b);

5) This can of course be extended to finite lists of such triples.

— encoding function F_{AV}: mapping from CI_{AV} to V_{AV}; encoding of AC_{AV} – entity structures: $F_{AV}(\langle m,\langle A_i,v_{ij}\rangle\rangle$ = <avContent xml:id=”c1” target=”#m” attribute= “$F_{AV}(A_i)$” value= “$F_{AV}(v_{ij})$”/>;

— **Semantics:** Interpretation function I_{AV} as defined in (D.4).

D.2.2 Plug-in for events and semantic roles

The following more general content plug-in is based on ISO 24617-4 (SemAF-SR) for annotating the semantic roles of the participants in an event. For example, the sentence *“The soprano sang an aria”* is annotated as in (D.6b), where “sing.01" refers to a verb sense in VerbNet.

(D.6) a. "The soprano sang an aria"

Markables: m1=“The soprano", m2=“sang", m3=“an aria"

b. <entity xml:id="x1" target="#m1" pred="soprano"/>
<event xml:id=”e1” target=”#m2” eventFrame=”sing.01” type =”accomplishment”/>
<srLink event=”#e1” participant=”#x1” semRole=”agent”/>
<entity xml:id=”x2” target=”#m1” pred=”aria”/>
<srLink event=”#e1” participant=”#x1” semRole=”theme”/>

SemAF-SR interprets such annotations as expressing the existence (or denied existence, in case of a clause with negative polarity) of certain states or events and participants in certain roles. For example, in (D.6) the semantics can be expressed by the following DRS.

(D.7) [e1, x1, x2 | sing01(e1), soprano(x1), aria(x2), agent(e1,x1), theme(e1,x2)]

The plug-in consists in this case of the abstract and concrete syntax of the SemAF-SR markup language and its semantic interpretation function, which produces DRSs like those in (D.7). The conceptual inventory of the abstract syntax lists semantic roles and verb senses by reference to VerbNet[37], defines entity structures for eventualities and their participants, and defines link structures for relating participants to eventualities in a certain role. The concrete syntax defines XML encodings as illustrated in (D.6b).

The simplest content plug-in for semantic roles is one that takes a minimalist approach to event classifications, and uses a simple form like `<event xml:id="e2" target="#m3" pred="sing"/>` rather than the more fine-grained representations of SemAF-SR or ISO-TimeML. This plug-in (`PL_{SR}') is characterized by the following schema:

— **Abstract syntax**: the conceptual inventory lists the semantic roles defined in ISO 24617-4 and a set of verb senses, distinguishing only senses, which differ in their semantic roles; entity structures are defined for eventualities and their participants, and link structures for indicating semantic roles;

— **Concrete syntax**: specifies names for the elements of the conceptual inventory; XML elements for encoding the entity and link structures;

— **Semantics**: translation of entity and link structures and their combination to DRSs.

Formally, this plug-in is defined as follows.

(D.8) $PL_{SR} = \langle\langle CI_{SR}, AC_{SR}\rangle, \langle V_{SR}, CC_{SR}, F_{SR}\rangle, \langle M_{SR}, I_{SR}\rangle\rangle$, with:

$CI_{SR} = SR \cup EP \cup DP$ made up of the set SR set of semantic roles defined in ISO 24617-4 (see Table D.1), a set EP of event predicates defined in VerbNet (such as ‘sing01’), and a set DP of domain predicates defined in a domain ontology (like ‘soprano’ and ‘aria’);

AC_{SR} = {entity structures $\langle m, p\rangle$; link structures $\langle e_{EV}, e_P, R\rangle$}6);

V_{SR} = names of CI_{SR} elements;

6) ‘eEV‘ stands for ‘event entity structure’, eP‘ for ‘participant entity structure’.

CC_{SR} = {<event>, <entity>, <srLink>};

F_{SR} assigns a value in V_{SR} to each element of CI_{SR}[7)];

$F_{SR}(\langle m, p\rangle)$ = <entity xml:id="x" target="#m1" pred=" $F_{SR}(p)$""/> if $p \in DP$;

$F_{SR}(\langle m,p\rangle)$ = <event xml:id="x" target="#m1" pred=" $F_{SR}(p)$"/> if $p \in EP$;

$F_{SR}(\langle e_{EV}, e_P, R\rangle)$ = <srLink event="$F_{SR}(e_{EV})$" participant="$F_{SR}(e_p)$" semRole="$F_{SR}(R)$"/>;

$I_{SR}(p) = p'$; $I_{SR}(t) = t'$; $I_{SR}(R) = R'$; $I_{SR}(\langle m,p\rangle) = [x \mid p'(x)]$;

$I_{SR}(\langle e_{EV}, e_P, R\rangle) = [\, e\ x \mid R'(e,x)) \,] \cup I_{SR}(e_{EV}) \cup I_{SR}(e_P)$.

The interface for using this plug-in is the same as the one defined in (D.2).

D.2.3 Plug-in for events, participants, and quantification

A plug-in for the semantic content of dialogue acts 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 that may express semantic content. On top of the identification of events with their time and 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 for this type of information[20],[23] can be the basis of a powerful future plug-in for this purpose.

Table D.1 — Semantic roles defined in ISO 24617-4

	Role	Definition
1.	Agent	Participant in an event who intentionally or consciously initiates an event, and who exists independently of the event.
2.	Beneficiary	Participant in an eventuality that is advantaged or disadvantaged by the eventuality, and that exists independently of the event.
3.	Cause	Participant in an event that initiates the event, but does not act with any intentionality or consciousness; the participant exists independently of the event.
4.	Goal	Participant in an event that is the (non-locative, non-temporal) end point of an action; the participant exists independently of the event.
5.	Instrument	Participant in an event that is manipulated by an agent, and with which an intentional act is performed; it exists independently of the event.
6.	Partner	Participant in an event that is intentionally or consciously involved in carrying out the event. Participant is not the principal agent of the event, and exists independently of the event.
7.	Patient	Participant in an event that undergoes a change of state, location or condition, is causally involved or directly affected by other participants, and exists independently of the event.
8.	Pivot	Participant in a state that is characterized as being in a certain position or condition throughout that state, and has a major or central role or effect in that state. A pivot is more central to the state than a participant in a theme role, and exists independently of the state.
9.	Purpose	Set of facts or circumstances that an agent wishes or intends to accomplish by performing some intentional action.
10.	Reason	Set of facts or circumstances explaining why a state exists or an event occurs.
11.	Result	Participant in an event that comes into existence through the event; it indicates a terminal point for the event: when that is reached, the event does not continue.
12.	Setting	Set of (non-locative and non-temporal) facts or circumstances of the occurrence of an event or a state.
13.	Source	Non-locative, non-temporal starting point of an event. The source exists independently of the event.

7) The value assigned to an argument a will be designated by a'.

Table D.1 *(continued)*

	Role	Definition
14.	Theme	Participant in a state or an event that (i) in the case of an event, is essential to the event taking place, but does not have control over the way the event occurs and is not structurally changed by the event, and (ii) in the case of a state, is characterized as being in a certain position or condition throughout the state, and is essential to the state being in effect but not as central to the state as a participant in a pivot role. The theme of a state or event exists independently of the state or event.
15.	Manner	The way or style of performing an action or the degree/strength of a cognitive or emotional state.
16.	Medium	The physical setting, device or channel that allows an event to take place.
17.	Means	Procedure for performing an action in terms of component steps, or a methodology by which an intentional act is performed by an agent. A means does not necessarily exist independently of the event.
18.	Location	Place where an event occurs, or a state is true, or a thing exists.
19.	Initial Location	Participant in an event that indicates the location where an event begins or a state becomes true; initial-location exists independently of the event.
20.	Final Location	Location where an event ends or a state becomes false; final-location exists independently of the event.
21.	Path	Intermediate location or trajectory between two locations, or in a designated space, where an event occurs.
22.	Distance	Length or extent of space that plays a role in an eventuality.
23.	Time	Participant that indicates an instant or a time interval during which a state exists or an event takes place.
24.	Duration	Length or extent of time during which an event occurs or a state is true.
25.	Initial Time	Indication of the point in time when an event begins or a state becomes true.
26.	Final Time	Indication of a point in time when an event ends or a state ceases to be true.
27.	Amount	Quantity of something other than time or space, or number of objects of a certain kind, which plays a role in an event or a state.
28.	Attribute	Property that an event or state associates with one of the other participants. ‘(We [agent e1]) will (paint e1) (the front door [theme e1]) (dark green [attribute e1])’

D.3 Plug-ins for rhetorical relations

This second edition of this document supports the marking up of rhetorical relations in dialogue in a more fine-grained way than the first edition, see 5.3, but does not specify any particular set of relations to be used. A plug-in for discourse relations does not require the introduction of any entity structures or link structures, since these have been defined in DiAML, and for the same reason no specification of an interface is required, but just the specification of a set of rhetorical relations and their argument roles.

The plug-in specified takes the ISO 24617-8 (DR-core) annotation scheme as the point of departure. The relation ‘Evaluation’ is added. Table D.2 lists the resulting 19 relations with their definitions, which describe their semantics in an informal way. Many other relations that are distinguished in other annotation schemes can be seen as special cases of DR-core relations, for example ‘Explanation’ as a case of ‘Cause’, ‘Juxtaposition’ as a case of ‘Contrast’, and ‘Specification’ as a case of ‘Elaboration’.

Table D.2 — Rhetorical relations, based on ISO 24617-8

	Relation	Definition
1.	Cause	The second argument provides a reason why the first argument occurs or holds true.
2.	Condition	The first argument is an unrealized situation which, when realized, would lead to the situation that forms the second argument.
3.	Negative Condition	The first argument is an unrealized situation which, when not realized, would lead to the situation that forms the second argument.

Table D.2 *(continued)*

	Relation	Definition
4.	Purpose	The second argument is the goal or purpose of the situation that forms the first argument.
5.	Manner	The second argument describes how the first argument comes about or occurs.
6.	Concession	The second argument cancels or denies an expected causal relation between the first argument and the negation of the second.
7.	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.
8.	Exception	The second argument indicates one or more circumstances in which the situation that forms the first argument does not hold.
9.	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.
10.	Substitution	The two arguments are alternatives, the situation of the second argument being the favoured or chosen alternative.
11.	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.
12.	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.
13.	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.
14.	Elaboration	The two arguments are the same situation, but the second argument is specified in more detail.
15.	Restatement	The two arguments are the same situation, but viewed from different perspectives.
16.	Synchrony	The two arguments form two temporally overlapping situations. All forms of overlap are included.
17.	Asynchrony	The first argument temporally precedes the second.
18.	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.
19.	Evaluation	The second argument provides an opinion on the social, aesthetic, economic, or other qualities of the first argument.

This leads to the following plug-in PL_{DR} for discourse relations in dialogue:

Abstract syntax:

— CI_{DR} = the relations defined in DR-core (Table D.2), and their argument roles, as specified in Table D.3.

Concrete syntax:

— XML names for the relations in the conceptual inventory and their argument roles. For convenience, font variations of the same strings are used: "cause", "condition",.. and "reason", "result", etc. (Not all relations have distinct argument roles. For example, *Contrast* and *Similarity* have two arguments that play semantically identical roles. In such cases, the arguments are named "arg1" and "arg2", and by convention the label "arg1" is given to the argument that occurs first in the discourse.)

Semantics:

— The meanings of the rhetorical relations and their argument roles are listed in Table D.2. See also A.4.3.3 for their use in the semantic interpretation of rhetorical link structures.

Table D.3 — Rhetorical relations and argument roles in the PL_{DR} concrete syntax

Rhetorical relation	First argument	Second argument
cause	reason	result
condition	antecedent	consequent
negativeCondition	negatedAntecedent	consequent
purpose	enablement	goal
manner	means	achievement
concession	expectationRaiser	expectationDenier
exception	regular	exclusion
substitution	disfavoredAlternative	favoredAlternative
exemplification	set	instance
elaboration	broad	specific
asynchrony	before	after
expansion	narrative	expander
evaluation	situation	judgement
contrast, conjunction, disjunction, restatement, similarity, synchrony	arg1	arg2

D.4 Plug-ins for additional communicative functions

D.4.1 Overview

A plug-in PL_{CF} for adding certain communicative functions can have a very simple specification, since no new entity structures or link structures are needed, but only the following components:

— **Abstract syntax:** conceptual inventory CI_{CF} listing the new functions;

— **Concrete syntax:** CV_{CF}: corresponding XML function names, and mapping from CI_{CF} to CV_{CF};

— **Semantics:** the context-update semantics $I_a(f_j)$ for every $f_j \in CI_{CF}$.

Since no new structures are introduced, no plug-in interface is required.

D.4.2 Plug-in for application-specific communicative functions

This document was designed to be domain-independent, and for this reason does not include communicative functions that would be specific for a certain application domain. All its communicative functions are either general-purpose or belong to one of the dialogue control dimensions. The general-purpose functions of the scheme form a powerful battery in any application, but some applications may benefit from the availability of additional, domain-specific communicative functions. This is another area where plug-ins can be useful.

When designing a plug-in for domain-specific dialogue acts, the question arising is how these functions relate to those of the host annotation scheme. Any plug-in should respect the orthogonality of the dimensions and the semantic connectedness of the communicative functions, see Clause 9. For example, in a negotiation domain one finds bids, counter-bids, accepts and rejects of (counter-) 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.

D.4.3 Plug-in for additional SOM functions

Most dialogue annotation schemes have been designed for task-oriented dialogues, where the participants have a clear objective such as finding a train connection or making an appointment.

Everyday conversations such as a chat with a neighbour or with a colleague at the coffee machine often do not have such a specific objective, but rather have a social purpose, such as establishing a pleasant atmosphere or maintaining a good relationship. Task-related dialogues often have an initial phase in which the participants engage in small talk before getting to a specific task. Such initial phases have often been omitted in dialogue corpora, where the initial small talk is regarded as occurring ‘before’ the ‘actual’ dialogue. An exception is the ADELE corpus of casual conversations in the form of textual chat dialogues[33]. The dialogues in this corpus often have rather elaborate initial phases with greetings and discussions of each other’s health, and a leavetaking phase with various kinds of farewell greetings and well-wishing. In order to annotate the communicative functions in such phases in a satisfactory way, the DIT++ annotation scheme (Release 5.2) includes several dimension-specific functions in the Social Obligations Management dimension[8]. Table D.4 lists these functions and gives some examples. These functions can be used to specify a plug-in PL_{SOM} as follows:

— **Abstract syntax:** conceptual inventory CI_{SOM} listing the new functions; see Table D.4;

— **Concrete syntax:** corresponding XML function names;

— **Semantics:** the context-update semantics $I_a(f_j)$ for every $f_j \in CI_{SOM}$.

Table D.4 — Additional communicative functions for Social Obligations Management (from DIT++)

Communicative Function	Definition	Example
Follow-on Greeting	The sender, S, wants the addressee, A, to know that S, has established the presence and the identity of the addressee, A.	“Hi Anne.”
Politeness Question	The sender, S, wants to know the state of well-being of the addressee, A, or of someone close to A.	“How do you do?” “How is your mother?”
Return Politeness Question	The sender, S, wants to know the state of well-being of the addressee, A, or of someone close to A; S responds to a Politeness Question by A.	“And how about you?”
Opening Politeness Statement	The sender, S, wants the addressee, A, to know that S is pleased to meet A.	“Nice to meet you”
Closing Politeness Statement	The sender, S, wants the addressee, 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	The sender, S, wishes the addressee, A, to be in a positive state of well-being, and intends to close the dialogue.	“Have a good time.”

D.4.4 Plug-in for additional feedback functions

For applications that instantiate use case UC4, i.e., for the generation of dialogue acts in a dialogue system, it is useful to have more fine-grained feedback functions available than the rather coarse positive and negative functions defined in the first edition of this document. More fine-grained functions are available in the DIT++ annotation scheme, where a distinction is made between five levels of processing at which feedback may be provided or elicited: (1) attention, (2) perception, (3) interpretation, (4) evaluation, and (5) execution. These functions define a plug-in PL_{FB} as follows:

— **Abstract syntax:** conceptual inventory CI_{FB} listing the new functions; see Table D.5 for the 10 fine-grained auto-feedback functions; similar tables define the allo-feedback functions (10 feedback-providing and 10 feedback eliciting functions);

— **Concrete syntax:** corresponding XML function names;

8) See Reference [62].

— **Semantics:** the context-update semantics $I_a(f_j)$ for every $f_j \in CI_{FB}$.

Table D.5 — Fine-grained communicative functions for auto-feedback (from DIT++)

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

D.5 Plug-ins for emotion annotation

The sender of a dialogue act may express a certain emotion associated with the performance of the dialogue act, such as amusement, irritation, or disappointment. The first edition of this document had no provisions for annotating this aspect. Qualifiers, as used for sentiment and certainty, can also be considered for dealing with a speaker's affective/emotional state, but this would carry the simplistic assumption that such states can be characterized in a one-dimensional way, through a single predicate.

The W3C recommendation EmotionML is a flexible scheme, designed for being combined with other annotation schemes. It characterizes 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'[9]. An emotion in EmotionML may have components of various categories; for example, in (D.9) an emotion is annotated as being a form of anger with elements of sadness and fear[10].

(D.9)
```
<emotion category-set="http://www.w3.org/TR/emotion-voc/xml#big6">
    <category name="sadness" value="0.3"/>
    <category name="anger" value="0.8"/>
    <category name="fear" value="0.3"/>
</emotion>
```

Observing that there is no general agreement in the community, EmotionML does not provide a single repository of emotion descriptors, but gives users a choice to select a suitable emotion vocabulary in their annotations. In order to promote interoperability, EmotionML provides a number of emotion

9) Emotion Markup Language (EmotionML) 1.0, W3C Recommendation 22 May 2014. Available at http://www.w3.org/TR/2014/REC-emotionml-20140522/. See also Reference[26].

10) Taken from the document https://www.w3.org/TR/emotionml/.

vocabularies that can be used for this purpose.[11)] The guiding principle for selecting emotion vocabularies has been to list vocabularies that are either commonly used in technological contexts, or represent current emotion models from the scientific literature. One of the best known repositories is Ekman's 'big six' [32].

EmotionML is defined only at the level of concrete syntax, so it cannot directly be used as a plug-in for ISO-compliant annotation. However, an abstract syntax for EmotionML can be developed using the CASCADES methodology in reverse engineering mode (see Reference [16]), and a semantics can be added for those parts of EmotionML markups that are truly semantic in nature, in contrast with e.g. confidence values.

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 (D.10) 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.

(D.10)a. P1: Would you like to have a cup of coffee? (= markable m1)

P2: That would be wonderful! (= markable m2)

b.
```
<dialogueAct xml:id="da1" target=#m1" speaker=#p1" addressee="#p2"
      dimension="social" communicativeFunction="offer"/>
<contentLink dialAct="#da1" content="#e1"/>
<dialogueAct xml:id="da2" target="#m2" speaker="#p2" addressee="#p1"
      dimension="social" 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"/>
```

A plug-in PL_e for adding emotion annotation to dialogue acts, based on EmotionML, can be defined as follows.

Abstract syntax:

— CI_e = set of emotion categories; intensity values (any floating point number in the interval [0,1];

— Entity structures: pairs ⟨m, ⟨c, v⟩⟩ consisting of an emotion category and an intensity value.

Concrete syntax:

— XML names for the emotion categories in CI_e; numerical values representing emotion intensities;

— encodings of entity structures using <emotion> elements (as defined in EmtionML, but simplified).

Semantics:

— Pairs ⟨c, v⟩ are interpreted as attribute-value pairs where c denotes a two-place function, applicable to the experiencer and the object of an emotion, so $I_e(\langle c, v\rangle) = \lambda x.\ \lambda y.\ I_e(c)(x,y) = I_e(v)$.

For linking emotion specifications to dialogue act annotations, a plug-in interface is needed that defines the <emoLink> element with its underlying abstract syntax and semantics. In the abstract syntax, an emotion link structure is a triple ⟨p, s, e⟩ formed by the sender of a dialogue a ('p'), the semantic content ('s'), and an emotion ('e'). These components are the values of the attributes @holder, @object, and @emotion in the concrete syntax, as illustrated in (D.13). The semantics of the emotion link structures is defined by (D.11), where ${}_{a+c}I_e$ designates the semantic interpretation function of the interface of

11) The EMotionML repositories are listed in the W3C document 'Vocabularies for EmotionML', available at https://www.w3.org/TR/emotion-voc/.

the emotion plug-in with the annotation scheme formed by the host scheme of this document with a semantic content plug-in L_c.

(D.11) $\;{}_{a+c}I_e(\langle p, s, e\rangle) = I_e(e)(I_a(p), I_c(s))$

Annex E
(informative)

Annotation guidelines and examples

E.1 Segmentation

Dialogue acts are expressed in *functional segments,* defined as *minimal* stretch of communicative behaviour that have a communicative function. The requirement of being 'minimal' ensures that communicative functions are assigned as accurately as possible to stretches of behaviour. Consider the following example.

(E.1) Can you tell me what time the train to *uh,...* Viareggio leaves?

Here a conditional request is interrupted by a segment with a *Stalling* function. The segmentation distinguishes in this case one discontinuous functional segment in the Task dimension, namely fs1 = "Can you tell me what time the train to Viareggio leaves?" and one in the Time Management dimension, namely fs2 = *"uh,...",* leading to the following representation in DiAML.

(E.2)
```
<dialogueAct xml:id="da1" target="#fs1" speaker="#s" addressee="#a" dimension="task"
      communicativeFunction="request" conditionality="conditional"/>
<dialogueAct xml:id="da2" target="#fs2" speaker="#s" addressee="#a"
      communicativeFunction="stalling" dimension="timeManagement"/>
```

It may also happen that a dialogue act corresponds to more than one turn, as in the following example, where the utterances in turns 1 and 3 together form an *Answer*.

(E.3) 1. A: There are two flights early in the morning, at 7.45 and at 8.15

2. B: Yes

3. A: and two more in the evening, at 7.15 and at 8.30

The utterances 1 and 3 in this example form a single discontinuous functional segment.

In an actual annotation process, the identification of functional segments and assignment of communicative functions go hand in hand. For annotating dialogue acts with a (auto- or allo-)feedback-specific function, or an OCM- or a PCM-specific function, the annotation also goes hand in hand with the identification of the relevant reference segments, as the dialogue segments that these acts provide or elicit information about.

E.2 Segment explicit and implicit, implied and indirect functions

The communicative function(s) of a dialogue segment may be recognisable in one of two ways: (1) through its linguistic or nonverbal properties which, in the context in which the segment occurs, are indicators of certain functions: or (2) through inference, when a function is implied by another function which the segment has (typically for the reason (1)). In the first case it is common to say that the segment has that communicative function *explicitly*; in the second case that it has that function *implicitly*. The following example illustrates this.

(E.4) 1. A: Would you like to have some coffee?

2. B: Some coffee would be great, thanks.

A's utterance may be understood as an *Offer*; due to the fact that expressions of the form "*Would you like…*" are commonly used for that purpose; B's response is an *Accept Offer* by virtue of its linguistic form and the fact that it occurs immediately after an *Offer*. Since an offer can only be accepted when it has been understood, B's response by implication also has a positive auto-feedback function.

A functional segment expressing a dialogue act DA1 which has a functional dependence relation to a previous dialogue act DA2, always has an implied auto-feedback function relating to the functional segment where DA2 was expressed. This is one important type of implicit functions that functional segments may have, and it is one of the sources of the multifunctionality of functional segments. More generally, the following types of implicit communicative function can be distinguished.

a) A communicative function F_2 is *logically entailed* by the communicative function F_1 if F_1 is a special case of F_2. This happens in hierarchies of communicative functions like the general-purpose functions defined in Annex C, where for instance a *Confirm* is a special case of an *Answer*, and a *Correction* is a special case of a *Disagreement*, which in turn is a special case of an *Inform*.

b) A communicative function F_1 may have another function F_2 as a *conversational implicature*, i.e., in most situations where a functional segment has the function F_1 it also has the function F_2, assuming that the dialogue participants behave cooperatively. For example, a thanking act like *"Thank you"* will normally be understood as being also a signal of positive feedback.

c) Turn-taking functions: Every time someone starts speaking, this can be interpreted as the performance of a turn-taking act; every time someone stops speaking, this can be interpreted as a turn-release act; and every time a speaker goes on speaking, this can be interpreted as a turn-keeping act. See E.3.2.

Should implicit communicative functions be annotated? Annotating logically entailed functions would be redundant, since by their very nature such functions can be inferred. For conversationally implicated functions the situation is different, since these functions cannot be inferred with certainty. The annotation of implicated positive feedback functions would be impractical; the annotation of implied turn management functions would even be impossible (see E.3.2). If the recognition of implicated functions is a focus of interest, then their annotation can be done post-hoc by checking whether the context allows the inference of these functions. When an annotator runs into a situation where a functional segment has an explicitly expressed communicative function and an implied one, should decide whether the implied function is a logical consequence or a matter of what is plausible in the given context. In the first case the implied function should not be annotated; in the second case it may. For more details about types of implicit functions and strategies for how to deal with them, see Reference [17].

Indirect speech acts are often regarded as just a polite *form* of the same speech act as the direct form. By contrast, this document takes the view that indirect forms signal subtly different packages of beliefs and intentions than direct ones, and thus express a slightly different communicative act. For example, the direct request *"Tell me what time it is please"* carries the assumption that the addressee knows what time it is, whereas an indirect question like *"Do you know what time it is?"* or *"Can you tell me what time it is?"* does not carry that assumption (or it does at least not *express* that assumption), and can be interpreted as the conditional request *"Please tell me what time it is, if you know/can"*. This is represented in DiAML as follows:

(E.5)
```
<dialogueAct xml:id="da1" target="#fs1" sender="#s" addressee="#a"
    dimension="task" communicativeFunction="request"
    conditionality="conditional"/>
```

E.3 DiAML-annotations

E.3.1 Dimensions

Once a functional segment has been identified, its annotation is the indication of the dialogue act(s) expressed in the segment. The most important part of this is to assign it a communicative function (possibly more than one), and a dimension (one for each communicative function). The choice of a dimension is determined by the type of semantic content of the dialogue act, and can be made by answering the question: Does it concern

1) advancing the underlying task/activity (dimension: Task); or

2) an aspect of performing the underlying task/activity (dimension: Task Management); or

3) the speaker's processing of previous utterances (dimension: Auto-Feedback); or

4) the addressee's processing of previous utterances (dimension: Allo-Feedback; or

5) the allocation of the speaker role (dimension: Turn Management); or

6) the time needed to continue the dialogue (dimension: Time Management); or

7) the editing of what the speaker is saying (dimension: Own Communication Management); or

8) the editing of what the addressee is currently saying (dimension: Partner Communication Management); or

9) the organization of the dialogue (dimension: Discourse Structuring); or

10) ensuring contact between speaker and addressee (dimension: Contact Management); or

11) social obligations (dimension: Social Obligations Management)?

E.3.2 General-purpose functions

All dialogue acts with an *information transfer function* have the purpose of making certain information available to the addressee or of obtaining certain information. The information to be obtained or made available can be of any kind, relating to the underlying task or activity, or relating to some aspect of the communication. In order to decide whether a functional segment has an information transfer function, an annotator should thus decide whether there is such a purpose. If so, then the subtrees of the Information-providing and Information-seeking functions in Figure 2 can be used as decision trees, going systematically top-down left-right through the functions, checking the conditions that distinguish each function from their mother and sister functions. Since the functions at one level in a subtree are mutually exclusive, at most one of them applies. If one is found that applies, then go down one level to the functions dominated by this function, and repeat the process. Keep doing this until hitting a level where none of the functions applies.

Action discussion functions have in common that their semantic content describes an action, possibly with specifications of manner or frequency. The actions can be of any kind: actions for moving the underlying task forward, or actions for managing some aspect of the interaction, or actions for dealing with social obligations.

The action discussion functions fall apart into Commissives and Directives, familiar from speech act theory. Commissive acts have in common that the sender expresses a commitment to perform an action, while directive acts are characterized by the sender having the goal that the addressee commits himself/herself to performing a certain action. In order to decide whether a functional segment has a commissive or a directive function, an annotator should decide whether the segment has the purpose of expressing or trying to impose a commitment. If so, the annotator can use the subtrees of Commissives and Directives (see Figure 2) as decision trees, in the same way as for choosing an information-transfer function.

E.3.3 Dimension-specific functions

E.3.3.1 Auto- and Allo-Feedback

Both auto- and allo-feedback providing functions are divided into positive and negative ones, depending on whether they report successful or unsuccessful processing of previous utterances. Feedback is very