
# MATE Deliverable D2.1
# MATE Dialogue Annotation Guidelines

## 8. January 2000

## Authors

Mengel, A., Dybkjaer, L., Garrido, J.M., Heid, U., Klein, M.,
Pirrelli, V., Poesio, M., Quazza, S., Schiffrin, A., Soria, C.

# Contents

I. Introduction
II. Level specific markup
- Prosody
- Morphosyntax
- Dialogue acts
- Coreference
- Communication problems
- Cross level issues

III. Annexes
- Annex A: General markup guidelines
- Annex B: Coding module reference
- Annex C: DTDs

# I. Introduction

This report contains a comprehensive collection of recommendations or guidelines for representing descriptive annotation of spoken dialogue material. Descriptive annotation includes any information that encodes linguistic data with respect to their physical, perceptual, or functional dimensions. Spoken dialogue material refers to any collection of spoken dialogue data (human-human, human-system, or human-human-system), including not only speech files but also e.g. logfiles or scenarios which are related to the spoken dialogues. Spoken dialogue annotation is the only area considered in this report, however this does not exclude that the recommendations may apply to other areas as well.

An in depth analysis of spoken dialogue data typically requires close inspection of multiple levels of description whereas, e.g., various kinds of exploitation of data requires access to only one parlicular level. MATE has looked at both the dimension of individual levels and the cross-level dimension. The individual levels addressed by MATE includes prosody, morpho-syntax, dialogue acts, co-reference, and communication problems. These levels as well as cross-level issues have been addressed within the same common framework to ensure a common approach across levels. This framework makes it easier for the annotator to move from one level to another and facilitates the use of the same set of software tools and the same interface look and feel, independently of the level in question. MATE has addressed fairly many levels which were all found to work within the proposed framework. It is therefore very likely that it will also work for any other level following the same approach.

This report is primarily aimed at people working in the area of markup of spoken dialogue corpora. It builds on a common standard framework in terms of a coding module (see below) at the conceptual level and an underlying representation in XML at the implementational level. For each level considered by MATE recommendations are provided on how to encode relevant phenomena, one or more best practice coding modules are provided and several examples are given. This should make it easy for a person from the target group to apply the coding modules for markup and to design his/her own coding module for a level following the MATE approach. The MATE workbench - asfotware also developed in the MATE project for the support of annotation work - comes with all the coding modules presented in this report built-in and supported and there is general support for adding and applying new coding modules.

## 1 Approach

The MATE common framework builds on the use of coding modules, cf. D1.2, which basically is a coding scheme with the following 10 items:

| # | Item | Example |
|---|---|---|
| 1. | Name of the module, including an acronym. | *Verbmobil dialogue acts [VM-DA].* |
| 2. | Coding purpose of the module. | *To code task-specific dialogue acts for Task T7.* |
| 3. | Coding level. | *Dialogue acts.* |
| 4. | The type of data source scoped by the module. | *Spoken dialogue corpora* |
| 5. | References to other modules, if any. For transcriptions, the reference is to a resource. | *Orthographic transcription module OTM2 + Prosody module PM3 + Semantics module SM5.* |
| 6. | A declaration of the markup elements and their attributes.<br>An element is a feature, or type of phenomenon, in the corpus for which a tag is being defined. | - |
| 7. | A supplementary informal description of the elements and their attributes, including:<br>a. Purpose of the element, its attributes, and their values.<br>b. Informal semantics describing how to interpret the element and attribute values.<br>c. Example of each element and attribute. | - |
| 8. | An example of the use of the elements and their attributes. | - |
| 9. | A coding procedure. | - |
| 10. | Creation nodes | - |

The descriptions given in this document allow a complete separation from the underlying machine representation for which MATE uses XML. The separation means that in principle one could decide to other formats than XML at the implementational level without affecting the coding module in any way.

In this document recommendations will be made that rely on a given markup language, XML, that has already found broad support. This is an important factor as the availability of parsers and other software enhances the integration of this proposal into existing environments.

Aiming at maximal user-friendliness, descriptive appropriateness, integration of information, reuse, and computational efficiency necessarily requires technical description, thus the focus of this document is not a discussion of theoretical concepts of the five levels only, but a proposal for the annotation of the phenomena of these levels, their relations, and the encoding of that information.

The guidelines and the markup schemes proposed in this document are intended to be flexible. The general framework described above was applied to the levels investigated for the MATE project. The level specific coding modules or coding schemes developed are designed to follow and thereby exemplify the usability of the markup framework. As these coding schemes are examples, they certainly reflect level specific theoretical considerations, but the claim of this document is not that the markup schemes proposed for individual linguistic levels are the only

theoretical conceptualizations of the levels that can be encoded in this markup framework. In contrast, they have been developed for the community to have best practice annotation schemes for the individual levels, cf. D1.1, which can be adapted to special needs by refinement, language specific additions or which can be seen as templates for other levels of description.

## 2 Structure

The following chapters are organized as follows:

Chapter 2 *Level specific markup* includes the level wise description of the markup for individual levels covered in MATE. At the beginning of chapter 2 there is an outline of the level internal structure of the chapters that has been used to make descriptions as compatible as possible.

Chapter 2 and its sub-sections are relevant for people who want to

- use one or more of the annotation schemes proposed here,
- adopt one or more of the annotation schemes for their own annotation task
- build an annotation scheme and want to learn from the approach proposed here,
- are interested in the mapping of theoretic descriptions into syntactical markup structures in general.

Chapter 3 *Annexes* provides information relevant for the implementation of the approach described. It is intended for people who need to

- understand special XML problems and the approaches followed in this project (Annex A)
- understand implications of the approach for software supporting the processing of the XML data (Annex A),
- want to refer to an overview of the coding modules (Annex B).
- write DTDs for annotation files of the levels described (Annex C).

# II. Level specific markup

This chapter is dedicated to level specific theoretical issues, recommendations for the tagging of phenomena, and the related tags. All the following chapters roughly use the common structure for the description of the coding guidelines outlined below.

### Coding Purpose

For each level there is an introduction which briefly states the scope and the application/purpose of the level, i.e. what phenomenological aspect of language or communication is described.

### Existing Schemes

This section gives a summary of the discussions in MATE Deliverable D1.1, reporting what schemes have been looked at and have been taken into account on the phenomenological level and in terms of markup. There is

- a list of schemes looked at
- a list of phenomena mentioned
- links (hrefs) to passages in D1.1

### Selected Scheme(s)

This section describes what exactly has been chosen as the scheme(s) to be used for the level. Either one or more existing schemes have been chosen, parts of different schemes have been selected, or a new scheme has been developed. In fact, none of the selected schemes existed in the description format of a coding module or corresponding XML representation before.

If more than one scheme has been selected - as in Dialogue Acts - the following structure would appear for each of the schemes described.

#### Scheme Name

In this sub-section the scheme is described in its overall structure:

- **Markup Declaration**

  Here, a hierarchical overview of the phenomena and their embedding structure is given (e.g., *phone*, *tone*, *break index*).

- **Description of Elements**

  For each phenomenon - which corresponds to an element - within the level (e.g., prosody) the following information is specified.

- **Phenomenon name**

  The name of the phenomenon to be tagged serves as a header of the chapter.

- **Description**

  The phenomenon (element) is described. The selected element name (e.g., `<word>`) is introduced and described in more or less detail according to the status of the theoretical discussion. Also, the properties of the phenomenon are described for the element.

- **Data Source**

  This section describes what kind of existing information is needed (i.e., what elements can be selected) in order to label phenomena of this type: One may need words for the markup of phrases, one needs utterances or sentences for the markup of dialogue acts etc.

- **Segmentation/Selection**

  What are the criteria important for the segmentation or selection process that that select one or more elements of the existing (base) level of annotation as starting point for the newly to be created element of this level, i.e. what are the selection criteria that make a given sequence of sounds a syllable? Note that in some cases there is a segmentation task. i.e. one segments an existing higher level phenomenon into smaller pieces, or there is a selection task where some elements are chosen to belong to a new kind of phenomenon.

- **Assignment**

  Criteria for selecting attribute values (properties) for the elements are discussed. The attributes and their value ranges are introduced. This information must be provided for each of the attributes a given element can have. Of course, the details given here may vary a lot. This information can also be provided by a kind of decision tree (cf. DAMSL).

- **Example**

  An example is given, starting with a data source example and describing segmented and labelled elements together with their attributes and values.

- **Coding Procedure**

  A recommendation on how to proceed when applying the markup to data. This may also include proposals for software or reliability checks etc. that have proven to be useful.

- **Markup Table**

  In the markup section the element name, the attributes and their values used will be summarized in a table like representation. Please note that there is not a one to one correspondence between the table contents and the underlying XML representation. The table below represents the grid of specification options. In the

upper row the element name is specified, in all rows following the element name, for each possible attribute of the given element, the attribute names and the values are listed. The value specification can be represented as the value type (a), as a value set (b) or - in the case of href attributes - as the element class pointed to (c).

| <element_name> | |
|---|---|
| attribute name | value range |
| (a) value type | |
| start | [FLOAT] |
| age | [INTEGER] |
| name | [ASCII] |
| (b) value set | |
| colour | red, green, blue |
| (c) href target | |
| href | <other_element> |

- **Integrated Example**

This section gives an example of the integrated use of all elements, attributes and values described within this coding scheme.

- **Coding Procedure**

In the individual descriptions of elements and attributes there was a description for recommendations of the coding procedure. In this section general coding procedure guidelines that apply to the whole coding scheme and all of the phenomena covered are provided.

The following chapters describe the level-wise annotation recommendations.

# Prosody

*Silvia Quazza & Juan Maria Garrido*

# 1. Coding Purpose

In this chapter we describe a framework for the annotation of speech corpora at the level of prosodic analysis. The scope of the Prosody Level includes phonetic transcription, intonation annotation and prosodic phrasing. The intended phenomena pertain to aspects of speech that are not explicitly represented in its orthographic transcription, which may be considered the starting point for the other linguistic annotation levels considered in MATE. So, the Prosody Level integrates the linguistic description of dialogues with information closer to their actual acoustic realization. The common reference to the speech signal allows to align prosodic annotations with orthographic transcription and higher linguistic levels, enabling cross level analyses.

## 1.1 Scope

Prosodic phenomena are specific to *spoken* language. They concern the way in which speech sounds are acoustically realized: how long they are, how high and how loud. Such acoustic modulations are used by human speakers to express a variety of linguistic or paralinguistic features, from stress and syntactic boundaries, to focus and emphasis or pragmatical and emotional attitudes. Linguistics and speech technology have approached prosody from a variety of points of view, so that a precise definition of the scope of prosodic research is not easy. A main distinction can be drawn between acoustic-phonetic analyses of prosody and more abstract, linguistic, phonological approaches.

Linguistically relevant prosodic events concur to express sentence structure: they highlight linguistic units by marking their boundaries and suggesting their function. Linguistic-phonological descriptions of prosody, usually identify a set of *prosodic units* (phonological units with a scope wider than a segment), and a set of *prosodic phenomena* which are 'superimposed' on these units. Prosodic units are the natural scope of prosodic events. Several types of prosodic units (differing mainly in their scope) have been proposed: paragraphs, sentences, intonation groups, intermediate groups, stress groups, feet, syllables, mora... Although prosody is by definition *suprasegmental*, prosodic analyses take often the phoneme as their minimal unit, where to measure rhythmical variations and locate intonation events. The family of prosodic phenomena includes the suprasegmental features of *intonation, stress, rhythm* and *speech rate*, whose variations are relevant to express the function of the different prosodic units: the prominent syllable in the word will be marked by stress, a falling intonation contour will mark the conclusion of a sentence, a faster speech rate and lower intonation characterize a parenthetical phrase...

Such prosodic features are physically realized in the speech chain in terms of variations of a set of acoustic parameters. Acoustic-phonetic analyses identify the following 'phonetic correlates of prosody': *fundamental frequency (f0), length changes in segmental duration, pauses, loudness, voice quality.*

Depending on the research purpose and point of view, prosodic phenomena can be marked in a speech corpus by simple diacritics in its orthographic transcription, or by labels classifying

intonation contours and unit boundaries according to some phonological theory, or by detailed measures of the acoustic-phonetic parameters.

We refer to D1.1 for a more detailed discussion concerning prosodic phenomena and their possible codings. What we state here are our minimal assumptions on the *scope* of prosodic coding:

- coding should take into account at least **segmental duration**, **pauses** and **intonation**
- it should consider the structuring role of prosody and provide means to delimit prosodic units by marking **phrase boundaries**
- finally, it should allow both detailed phenomenological descriptions and more abstract functional ones, providing distinct levels for phonetic and phonological annotation.

# 2. Existing Schemes

Coding prosody appears as a complex task, which has to deal with the intrinsic complexity of prosodic phenomena and with the variety of purposes, theories and points of view from which prosody can be approached. Such complexity is reflected in the wide variety of existing schemes that can be found in the literature. Examples of coding schemes more or less explicitly inspired by the different intonation theories and approaches are reviewed in D1.1. The review, by no means exhaustive, gives brief descriptions of the following schemes:

PROSPA
IPA
TEI
ToBI
SAMPA
SAMPROSA
INTSINT
SAMSINT
IPO
TSM
TILT
VERBMOBIL
KIM
PROZODIAG (Lund)
GOETEBORG

These schemes have revealed differences both in the covered phenomena, and in the underlying theoretical assuptions. They reflect the different purposes of prosodic analysis, which go from the phenomenological description of prosody in itself, to the study of its relations with discourse structure and to its applications in speech technology - synthesis, recognition and dialogue systems. As stated in D1.1:

> "*Each experimental study has adopted some kind of prosodic representation suited to its purposes, from abstract labels to acoustic measures*".

The schemes range from simple diacritic symbols integrating the orthographic transcription of corpora intended for linguistic analyses (e.g. TEI, Göteborg...), to theory-dependent phonological labels for intonation contours and phrasing (e.g. ToBI), to phonetic-acoustic representation of the f0 curve (e.g. INTSINT, IPO, TILT, ...).

The conclusion in D1.1 is that defining a unique 'standard' coding scheme by choosing a single prosody annotation scheme seems a difficult task at this moment. Although, in the era of large speech corpora, there is a definite need for a common notation allowing for easy data exchange and comparison, a single scheme would certainly dissatisfy some of the many points of view in the field, would be unsuitable to some of the intended purposes, would be too detailed or too poor, too theoretically committed or lost in insignificant details.

# 3. The MATE 'meta-scheme' for prosody annotation

## 3.1 The 'meta-scheme'

Due to the variety of points of view in prosodic studies and the difficulty in selecting the most representative coding schemes, the MATE proposal for the Prosody Level offers a 'meta-scheme', a framework where different existing notation conventions can be integrated and possibly new ones can be developed. The framework is detailed enough to suit the richer phonetic and phonological schemes and flexible enough to admit partial filling of its structure and to allow for different schemes to cooperate.

Its definition reflects the multi-level nature of prosodic research - the fact that prosody can be studied both with a phonetic and a phonological approach - and the useful distinction between prosodic units and prosodic phenomena.

The MATE 'meta-scheme' for prosody is a **four-layer annotation structure,** in which the different elements discussed in D1.1 can be accommodated. The sublevels are the following:

1) **phonetic transcription**

conceived for the representation of phonetic segments (the 'phones'), but also of other phenomena related to the segmental aspects of prosody, such as pauses, and other sub-word units such as syllables

2) **phonetic representation of intonation**

intended for the phonetic annotation of intonation phenomena, where the shape of fundamental frequency curves (and possibly of other acoustic correlates of intonation, such as energy, which at present are not included in the meta-scheme) is described in detail, by means of stylization and/or explicit labelling

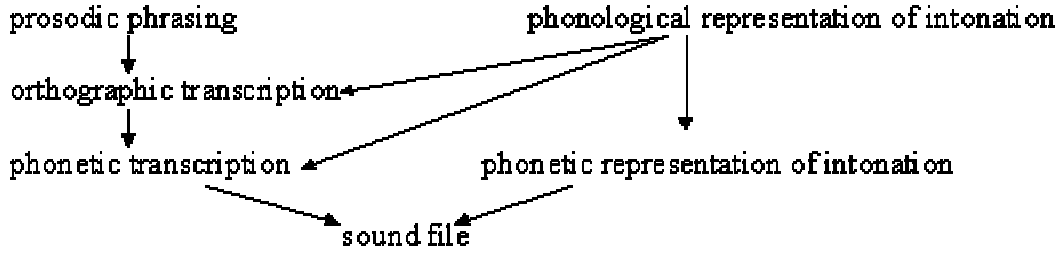
3) **phonological representation of intonation**

reserved for those schemes which annotate intonation from a phonological point of view, in terms of functional or underlying representations, and mark the role of relevant intonation events with respect to prosodic units

4) **prosodic phrasing**

intended for the segmentation of utterances in terms of high-level prosodic units (tone units, intonation groups, etc.)

The four levels do not represent a fixed hierarchy. The two phonetic levels, intended for phoneme segmentation and f0 description, are directly aligned with the speech signal and in this sense may be considered as *base levels*. The two phonological levels, describing the linguistically relevant intonation events and the prosodic structure of the utterance, keep a natural relationship both with the base prosodic levels and with other linguistic units. So, different links can be established between levels. It is conceivable to associate an intonation event such as "pitch accent" or a "boundary contour" to the word or phrase (orthographic level) on which it occurs as well as to the syllable or vowel on which it reaches its f0 target (phonetic transcription level) or to the corresponding configuration of pitch movements (phonetic description of f0). The following picture sketches the possible links between levels:




In the actual use of the scheme, the levels and their links can be fully or partially specified. In a linguistic text-oriented analysis, prosody could be considered in its function, leaving out the details of its realization. In this case, the sole phonological levels may be filled and linked to the orthographic level of words. Complex schemes like ToBI ([Silverman *et al.*, 1992], D1.1A) could be used in this way, or simpler schemes providing labels to distinguish types of accents, associated with words, and types of intonation boundaries.

In a speech technology context, a more signal-oriented approach could be adopted. In order to recognize or synthesize prosodic patterns, detailed phonetic descriptions are necessary, requiring both phonetic segmentation and phonetic representation of intonation - in terms of pitch movements or target f0 levels. The annotator would in this case look at the signal to segment it and possibly stylize its f0 profile and accurately label the stylized curve. For a complete analysis, he would link the detected units and events - phonemes and f0 variations - to the phonological descriptions of intonation contours and phrase structure.

## 3.2 Instances of the ’meta-scheme’

The first goal of the MATE ‘meta-scheme’ is then to provide an empty framework where the existing (or future) prosody annotation schemes could be represented in a common (and accordingly compatible and easy-to-compare) format. But it has been also conceived to allow the annotation of corpora using some of the most widely used existing annotation schemes. For each layer, (at least) one existing coding scheme has been adapted to XML, in order to be integrated within the MATE workbench and provide both a ready-to-use instance of the meta-scheme and an example and guideline for future adaptation of other schemes.

The chosen schemes for each level are the following:

1) phonetic transcription: SAMPA ([Wells *et al.* 1992]; D1.1A )

2) phonetic representation of intonation: INTSINT ([Hirst, 1991, 1994; Hirst & Di Cristo, 1998]; D1.1A), IPO ([t'Hart *et al.*, 1990]; D1.1A )

3) phonological representation of intonation: ToBI ('Tones'layer) ([Silverman *et al.* 1992]; D1.1A )

4) prosodic phrasing: ToBI ('Break-Indices' layer)

Widespread schemes have been preferred as examples. In the case of phonetic description of intonation, two schemes have been selected in order to represent both the 'pitch movement' approach and the 'target level' one. It should be noted that for some schemes a reference definition is available, although not so strictly respected in actual applications (ToBI has a number of 'variants' and is subject to language-adaptation). For IPO, the reference is the classical text in which the methodology of perceptual analysis of intonation has been proposed, which was not explicitly intended to define a notation system. In any case, some simplifications or additions to the original schemes have been performed, in order to obtain a coherent adaptation.

As suggested above, each scheme can be used alone or can be integrated with the others. One could for example keep with IPO methodology and use SAMPA for phoneme segmentation and IPO for f0 description (and possibly a newly defined IPO-like "pitch configuration" scheme for the phonological level...). Or integrate the four layers using SAMPA, INTSINT and ToBI. To allow such modular approach, separate DTD's have been defined for each pair layer:scheme. These DTD's are included in the Annex.

The elements and attributes identified in the selected schemes are described in detail in the following. It should be noted that level 2), both in its IPO and INTSINT instances, has an inner structure corresponding to a typical three-step procedure in the phonetic annotation of intonation: obtain the raw f0 curve (element `<f0>`), stylize it (elements `<closecopy>` and `<momel>`) and label it (elements `<pitmove>` and `<intone>`). At the phonetic segmentation level, a useful extension is the `<syllable>` element, to which the element `<phone>` can be subordinated and to which the phonological intonation labels could profitably be linked. For each of the other levels a single main element is defined: `<tobitone>` for level 3) (`<target>`, `<f0range>`, and `<repair>` are accessory information), `<breakindex>` for level 4).

The list of elements adapted to XML, which is accordingly available for use in the MATE workbench, is the following:

1) Phonetic transcription

`<syllable>`

`<phone>`

2) Phonetic representation of intonation

`<f0>`

`<closecopy>` (IPO)

`<pitmove>` (IPO)

`<momel>` (INTSINT)

`<intone>` (INTSINT)

3) Phonological representation of intonation

`<tobitone>`

<target>

<f0range>

<repair>

4) Prosodic phrasing

<breakindex>

In the following, each pair layer:scheme will be described in a separate paragraph. For layer 2, to avoid duplication of descriptions, a single description will be given of the element <f0> for the raw f0 curve, that is present in both schemes IPO and INTSINT. Moreover, it should be noted that there is apparently no formal difference between the respective elements for the stylized curve <closecopy> and <momel>, both consisting in target points on the f0 curve. The substantial difference is in the intended interpolation function between the target points, which is linear for <closecopy> and parabolic for <momel>, and in the intended stylization procedure (manual vs. automatic).

# 4. Layer 1: Phonetic Transcription - SAMPA scheme

## 4.1 Markup Declaration

The layer of phonetic transcription is a base level intended for the representation of the minimal units for phonetic and prosodic analysis: phones and syllables. The level defines a base element, the <phone> element, corresponding to a segment in the speech signal, labeled according to its phonetic features. A <syllable> element may be added, consisting of a sequence of <phones>. The annotation at this level is a *transcription* and a *segmentation*, in the sense that it refers directly to the speech signal, recognizes the uttered sounds and splits the speech continuum into phonetic chunks. Each <phone> will then be classified with a phonetic label and associated with time information specifying its start and end instants. Higher linguistic levels, like the phonological prosodic levels or the orthographic word level, might inherit time information from the phonetic level by linking their elements with <phone> elements or <syllable> elements.

The scheme adopted here for phonetic transcription is SAMPA [Wells *et al.*, 1992], which is intended for multi-lingual phonetic transcription. In the original SAMPA notation, a transcription is a stream of phonetic labels and diacritics, where labels classify phones and diacritics give further specifications about phones, with the exception of stress marks which implicitly refer to the following syllable. In our adaptation, the <syllable> element is made explicit as a second layer built on top of the <phone> layer.

## 4.2 The <phone> element

### 4.2.1 Description

For the annotation of phones, SAMPA (SAM Phonetic Alphabet) has been chosen, providing a multilingual and computer-readable inventory of phonetic symbols.

The transcription task using SAMPA involves the use of a set of symbols and diacritics, which can be combined to represent the phonetic realisation of phones.

The considered SAMPA symbols provide labels for vowels and consonants. A further symbol (taken from the SAMPROSA extension of the SAMPA scheme) is considered for pauses, which

are marked as a special kind of sounds. Symbols can be combined together in some cases, e.g. two vowel symbols may be combined to represent diphthongs. The set of allowable combination may be language-dependent. A few diacritics are also available to mark additional features of phones: e.g. the length mark ":" may follow a phonetic label. The base symbols are listed below:

a) consonants

| IPA symbol | SAMPA symbol | phonetic description |
|---|---|---|
| b | b | voiced bilabial plosive |
| c | c | voiceless palatal plosive |
| ç | C | voiceless palatal fricative |
| d | d | voiced dental/alveolar plosive |
| ð | D | voiced dental fricative |
| f | f | voiceless labiodental fricative |
| g | g | voiced velar plosive |
| ɣ | G | voiced velar fricative |
| h | h | voiceless glottal fricative |
| j | j | palatal approximant |
| k | k | voiceless velar plosive |
| l | l | dental/alveolar lateral approximant |
| ʎ | L | palatal lateral appoximant |
| m | m | bilabial nasal |
| n | n | palatal nasal |
| ɲ | J | palatal nasal |
| ŋ | N | velar nasal |
| p | p | voiceless bilabial plosive |
| r | r | alveolar trill |
| ʁ | R | uvular trill/fricative |
| s | s | voiceless alveolar fricative |
| ʃ | S | voiceless postalveolar fricative |
| t | t | voiceless dental/alveolar plosive |

| | | |
|---|---|---|
| θ | T | voiceless dental fricative |
| ʋ | v | voiced labiodental fricative |
| w | w | labial-velar approximant |
| x | x | voiceless velar fricative |
| ɥ | H | labial-palatal approximant |
| z | z | voiced alveolar fricative |
| ʒ | Z | voiced postalveolar fricative |
| ʔ | ? | stod, glotal stop |

b) vowels

| IPA symbol | SAMPA symbol | phonetic description |
|---|---|---|
| a | a | open front unrounded |
| ɑ | A | open back unrounded |
| æ | { | near-open front unrounded |
| ɐ | 6 | near-open central unrounded |
| ɒ | Q | open back rounded |
| ɔ | O | open-mid back rounded |
| e | e | close-mid front unrounded |
| ɛ | E | open-mid front unrounded |
| ə | @ | mid central unrounded (schwa) |
| ɜ | 3 | mid central unrounded |
| i | i | close front unrounded |
| ɪ | I | near-close front unrounded lax |
| o | o | close-mid back rounded |
| ø | 2 | close-mid front rounded |
| œ | 9 | open-mid front rounded |
| ɶ | & | open front rounded |

| | | |
|---|---|---|
| u_ | u | close back rounded |
| U_ | U | near-close back rounded lax |
| }_ | } | close central rounded |
| A_ | V | open-mid back unrounded |
| y_ | y | close front rounded |
| Y_ | Y | near-close front rounded lax |

c) pause

| SAMPA (SAMPROSA) symbol | phonetic description |
|---|---|
| ... | silent pause |

The following SAMPA diacritics may be combined with the phonetic label (preceding or following it, according to the syntax suggested by the example):

| SAMPA symbol | phonetic description | Example of use |
|---|---|---|
| ~ | nasalization | O~ |
| = | syllabic consonant | =n |
| : | length mark | a: |

The user is referred to Wells *et al.* (1992) for a detailed description of the SAMPA symbols and their corresponding usage. More information is also available at 'http://www.phon.ucl.ac.uk/home/sampa/home.htm', including guidelines for the use use of SAMPA for transcription in the following languages: Bulgarian, Croatian, Danish, Dutch, English, Estonian, French, German, Greek, Hungarian, Italian, Norwegian, Polish, Portuguese, Romanian, Russian, Slovenian, Spanish and Swedish. A description of the SAMPROSA scheme can be found at 'http://www.phon.ucl.ac.uk/home/sampa/samprosa.htm'.

### 4.2.2 Data Source

Phonetic transcription is usually carried out from speech files, where the speech sound is sampled. Speech files can be listened to and graphically displayed on a time axis, so that phones can easily be time-aligned to sound.

### 4.2.3 Segmentation/selection

Phonetic transcription is a segmentation task: the speech sound is segmented into a sequence of adjacent chunks, each corresponding to a <phone>. While in principle one could listen to the recorded speech and write down the perceived phones and the corresponding time (measured by a clock...), a reasonable segmentation procedure should rely on sampled speech, graphically shown as a waveform on a time axis and possibly also displayed in its spectrographic representation. The annotator would select a signal portion on the screen, listen to it and inspect its shape. Each phone will be characterized by its peculiar shape and show two transition zones where the boundaries with the adjacent phones should be placed. On this basis the annotator would recognize the uttered phone and segment it, possibly by mouse clicking on its start and end point on the screen.

### 4.2.4. Assignment

The attributes considered here for the <phone> element are the following:

- **type**: label specifying the type of phone according to its phonetic classification. The phone is recognized by listening to the correponding sound and looking (if necessary) at the available signal representations; it is then classified according to the phonological system of the language and represented by the corresponding SAMPA label. The list of SAMPA base symbols and diacritics is given above. Symbols may be combined together or with diacritics into complex labels.

- **start**: time start of the phone, expressed in milliseconds from the beginning of the sound file. Time start of a phone will generally coincide with time-end of the preceding one, as phonetic transcription is a *segmentation* of the speech signal. The exact determination of the boundary between adjacent phones is somewhat arbitrary, as the articulatory and acoustic transition between sounds is smooth. For each class of phones a set of conventions can be set to prescribe where to place the boundary, depending on the available signal representations (waveform, spectrogram, etc.). A consistent application of explicitly stated segmentation criteria is recommended.

- **end**: time end of the phone, in milliseconds from the beginning of the sound file.

## 4.3 The `<syllable>` element

### 4.3.1 Description

In many prosodic descriptions the syllable is taken as the minimal prosodic unit, the building block of the rhythmical structure and the scope of intonation events. Formally, it is a sequence of one or more phonemes centered on a vocalic nucleus. Its precise definition is language and theory dependent. In SAMPA, the diacritics for primary and secondary stress are inserted at the beginning of the stressed syllable: e.g. ["meZ@] (*measure*), [@"nVD@] (*another*). So, even if the prosodic extension of SAMPA (SAMPROSA [Gibbon, 1989]) is not taken into account, the

notion of syllable is implicit in SAMPA phonetic notation. Here an element <syllable> is defined explicitly, linked to its component <phone>'s and possibly carrying the stress mark, according to the following definition:

| | |
|---|---|
| " | primary stress |
| % | secondary stress |

The SAMPA primary stress symbol (") can not be used in XML markup. For this reason, it has to be represented by "&quot".

### 4.3.2 Data Source

Syllables are defined starting from <phone>'s.

### 4.3.3 Segmentation/selection

After the phonetic transcription has been obtained, syllables are defined by selecting their component phones, from syllable boundary to syllable boundary, according to the phonetic syllabification rules of the language (and of the chosen linguistic theory), and judged as to its accent degree. Language- and theory-dependent automatic procedures could be implemented for syllabification.

### 4.3.4. Assignment

The attributes considered here for the <syllable> element are the following:

- **stress**: optional label specifying if the syllable is stressed, with primary (") or secondary (%) stress; if not specified, the syllable is unstressed
- **href**: a sequence of <phone> elements
- **start**: start of the first phone of the syllable, inherited from the first <phone> element
- **end**: end of the last phone of the syllable, inherited from the last <phone> element

## 4.4 Examples

The following example shows the phonetic transcription of the Spanish word *'casa'* ('house') and its corresponding syllabic segmentation, using the <phone> and <syllable> elements:

phone.xml

```
<phone id="phn_01" type="k" start="345" end="390"/>
<phone id="phn_02" type="a" start="390" end="450"/>
<phone id="phn_03" type="s" start="450" end="490"/>
<phone id="phn_04" type="a" start="490" end="540"/>
```

syllable.xml

```
<syllable id="sllbl_01" stress="&quot" href="phone.xml# id(phn_001)..id(phn_002)"/>
<syllable id="sllbl_02"               href="phone.xml# id(phn_003)..id(phn_004)"/>
```

## 4.5 Coding Procedure

Manual phonetic segmentation would be helped by a software tool displaying the speech signal in its waveform and spectrographic representations, allowing listening, selecting signal portions, zooming, selecting pre-defined phonetic labels, choosing segmentation points on the time axis. The set of allowable phonetic labels (for the given language) should be defined in the DTD (the DTD included in the Annex does not define language-dependent symbol sets), while a specific coding guideline document will explicitly state the adopted set of segmentation criteria. The coding procedure would then be:

1. select the speech file and open the synchronized windows for phonetic segmentation and waveform and spectrum display

2. zoom until a detailed inspection of the signal is possible

3. inspect and listen to the signal portion until the uttered phonemes are recognized

4. select a phonetic label for the first phone

5. identify its boundaries according to the segmentation criteria and mark them by placing the cursor on the proper point on the time-axis (this should automatically set the time attribute)

6. after phonetic segmentation is concluded, define syllables by selecting their component <phone>'s and, if stressed, by assigning the proper stress mark

Tools for automatic segmentation are available, often language dependent. Good performances are offered by phonetic aligners, that align a speech signal to a predefined phonetic transcription.

The procedure in this case would be:

1. listen to the speech sound and transcribe it as a sequence of phones

2. apply the phonetic aligner to the speech signal with its phonetic transcription and obtain its phonetic segmentation

3. import the phonetic segmentation in the MATE environment

4. define syllables as in step 6 above.

### 4.6 Markup Table

| <phone> | |
|---|---|
| id | [ASCII] |
| type | b,c,..,a,A,..,=,: |
| start | [FLOAT] |
| end | [FLOAT] |

| <syllable> | |
|---|---|
| id | [ASCII] |
| stress | ", % |
| start | [FLOAT] |
| end | [FLOAT] |

# 5. Layer 2: Phonetic Representation of Intonation

The phonetic representation of intonation should provide a detailed description of the utterance intonation profile, which is one of the main acoustic correlates of prosodic structure. The object of the description is fundamental frequency - an acoustic parameter which is calculated from the voiced portions of the speech signal by means of signal processing algorithms. Once the gaps of unvoiced phones have been interpolated, f0 is a continuous curve showing perceptually irrelevant variations, micro-prosodic variations due to phoneme quality and macro-prosodic variations which may have a linguistic function. A phonetic representation of this curve will ignore minor details but will describe the shape of the curve by classifying all its relevant features, that a functional phonological analysis could later interpret.

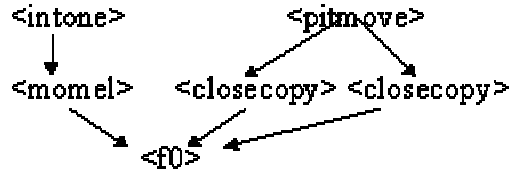
While there are relevant approaches (i.e. Fujisaki) describing intonation as a superposition of mathematically defined curves, here we consider the family of linear models representing f0 as a sequence of phonetic events. Two steps are necessary to obtain a phonetic representation of intonation:

- a stylization of the f0 curve, where irrelevant details (and possible errors of the pitch tracking algorithm) are removed and the curve is represented by a sequence of discrete elements: inflection points, interpolated with a linear or parabolic function

- a classification of the elements of the stylized curve

The elements of the stylized curve are the 'relevant variations' of f0. Depending on the point of view and the underlying intonation theory, such variations may be seen in their *movement*

between two f0 values or in their *target* value. So, you may see the curve as a chain of rises and falls or as a sequence of high and low values. The two approaches are represented by the two schemes that we have chosen as examples for layer 2. Both schemes start from the raw f0 curve (automatically obtained from the signal), represented as a sequence of frame by frame f0 values (<f0>). Both schemes rely on a stylization of the f0 curve, represented as a sequence of inflection points on the curve (named <closecopy> for IPO and <momel> for INTSINT, just to keep track of the different interpolation laws suggested by the two schemes). But the INTSINT description of the curve will directly label the inflection points as *target tones*, while IPO will label *pitch movements* from one inflection point to the following one. The difference will be reflected in the different use of the href attribute for the elements <intone> and <pitmove>, pointing to a single stylized element in one case and to two consecutive elements in the other.




In the following, the two schemes will be described separately. The description of the base <f0> element will be given once, as the element and its use are common to the two schemes. It should be noted that when the stylized curve is imported as such (obtained outside the MATE workbench), in its <closecopy> or <momel> version, it could be the base reference for prosodic annotation and the <f0> element may be unnecessary.

## 5.1. Layer 2: Phonetic Representation of Intonation - f0 contours

### 5.1.1 Markup declaration

Fundamental frequency (pitch) is a parameter estimated from the acoustic signal, in its voiced (quasi-periodic) portions. It is defined as the inverse of period length and generally measured in Hz (number of periods per second). Period length could in principle be manually measured on the waveform, but it is usually estimated by pitch detection algorithms, whose output can be the series of points in time corresponding to period boundaries or, more often, a sequence of pairs [time interval : f0 value], where f0 is the average fundamental frequency measured on the time interval or *frame* (typically a few milliseconds).

Here we define an element to represent such raw f0 values, whose sequence provides the so-called f0 contour of the utterance. It should be noted that pitch estimation algorithms are not fully reliable, so that raw f0 values should be considered just a starting point of intonation analysis rather than its unquestionable objective reference.

### 5.1.2 The <f0> element

#### 5.1.2.1 Description

This element has been included to allow each f0 value of an f0 contour to be considered as an XML element (and accordingly handled and displayed). Each <f0> element is intended to represent a pair [time interval : f0 value] of a f0 contour. The most useful representation of the <f0> element is a graphical display of the sequence of its values as a function of time (the f0 curve or contour or profile).

#### 5.1.2.2 Data Source

The f0 contour is computed directly from the speech signal file (although some pitch detection algorithms rely on phonetic segmentation to obtain better estimates of fundamental frequency).

#### 5.1.2.3 Segmentation/selection

<f0> elements will be generated automatically, from f0 values calculated by an f0 estimation algorithm. If possible, such an algorithm will be available in the workbench. Otherwise, the f0 values will be imported from external files.

#### 5.1.2.4 Assignment

The attributes considered here for the <f0> element are the following:

- **value**: the f0 value (in Hz)
- **start**: time start of the calculation frame
- **end**: time end of the calculation frame

### 5.1.3 Markup Table

| <f0> | |
|---|---|
| id | [ASCII] |
| value | [FLOAT] |
| start | [FLOAT] |
| end | [FLOAT] |

## 5.2 Layer 2: Phonetic Representation of Intonation - IPO scheme

### 5.2.1 Markup Declaration

The IPO methodology for the analysis of intonation relies on two main assumptions: the first is that what is not perceived is irrelevant for a linguistic description of intonation, the second is that we perceive tone variations (rise/fall movements) rather than tone levels (high/low). The steps in the perceptual analysis of intonation are:

1. obtain a stylized close copy of the original f0 curve, by approximating the original values with a sequence of straight segments: the re-synthesized signal should be perceptually equivalent to the original one

2. classify the f0 segments as pitch movements, according to their shape and position in the phone chain (the proper reference is the syllable)

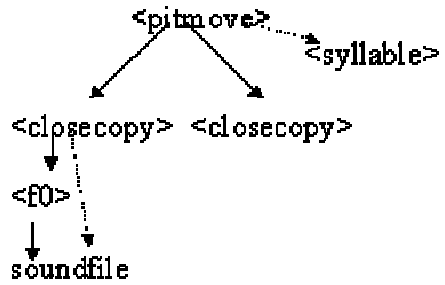
3. build up a grammar of admissible configurations of pitch movements and link intonation patterns to linguistic functions

Here we consider only the first two steps, which pertain to the phonetic representation of intonation. In order to represent them, we need three hierarchically ordered elements:

- `<f0>`, representing the points of the raw f0 curve
- `<closecopy>`, representing the inflection points in the stylized curve
- `<pitmove>`, representing the classified movements from one inflection point to the next one.

In principle, `<f0>` should be linked to the signal, `<closecopy>` to one `<f0>` element, and `<pitmove>` to two consecutive `<closecopy>` elements.

In actual annotation, it is not required that `<closecopy>` points coincide with `<f0>` points (a good stylization removes irrelevant excursions and possible pitch detection errors). Moreover, as suggested in the paragraph on coding procedures, if stylization is performed outside the MATE workbench, it could be directly imported, without reference to `<f0>`. In this case, the element `<closecopy>` will directly be aligned with the soundfile by means of its time attributes. Viceversa, a very simplified stylization (without the feedback of resynthesis), could be performed by directly linking `<pitmove>` to a sequence of `<f0>` elements, which could be thought of as approximated by a straight line. A further otpion would be to link the `<pitmove>` to the corresponding `<syllable>`: in this case, some of the precise acoustic content of the `<pitmove>` will be lost.




### 5.2.2 The `<closecopy>` element

#### 5.2.2.1 Description

This element has been included to allow each f0 inflection point of a ‘close copy’ stylised f0 contour (used in the IPO annotation system as the phonetic base representation of f0 contours) to be considered – and accordingly handled – as an XML element. The close copy is intended as a clean version of the f0 curve, where errors and irrelevant details have been removed, gaps corresponding to unvoiced phonemes have been filled and only the relevant movements are apparent. A more detailed desription of the concept of‘close copy’ can be found in ‘t Hart *et al.* (1990), among others. Such stylized description of f0 as a function of time can be displayed as a

sequence of straight segments connecting the relevant f0 values (inflection points), which may coincide with selected points (frames) in the raw f0 curve or simply approximate them.

#### 5.2.2.2 Data Source

The starting point for the creation of a close copy should be the raw f0 contour, together with the whole speech signal to allow for resynthesis. Phonetic segmentation would be useful as accessory information. In the MATE workbench, the close copy will most probably be imported from external files.

#### 5.2.2.3 Segmentation/selection

In the IPO methodology, the 'close copy' stylisations are defined by a resynthesis method which allows the perceptual definition of the relevant inflection points. The raw f0 curve is displayed, if possible aligned with phonetic segmentation. On this basis, the annotator draws a simplified curve which approximates the original one. He then listens to the speech resynthesized with the stylized artificial f0 values. He repeats these steps until he reaches the simplest stylization perceptually equivalent to the original contour.

The MATE workbench will not provide this complex environment. As a consequence, close copies will be imported from external files or will be obtained by a simplified (non-theory-conformant) procedure, where inflection points are directly selected on the raw f0 with no resynthesis feedback.

#### 5.2.2.4 Assignment

The attributes considered here for the <closecopy> element are the following:

- **value**: the stylized f0 value (in Hz) at the inflection point
- **href**: optional, points to an `<f0>` element
- **start**: time start of the stylised point
- **end**: time end of the stylised point

If the close copy is imported from an external file, the (link with the) `<f0>` element may not be necessary. In this case, as each inflection point is indeed a point, the two time attributes will have the same value. Alternatively, in case the close copy is obtained by selection of `<f0>` elements, href will point to the selected `<f0>`, from which the time attributes (and possibly the value) might be inherited, and consequently the two time values will be different (the first one corresponding to the beginning of the f0 calculation frame and the second one corresponding to the end of the frame).

### 5.2.3 The `<pitmove>` element

#### 5.2.3.1 Description

The element <pitmove> is intended for phonetic transcription of intonation contours according to IPO methodology. Whithin the IPO framework, a 'pitch movement' is a portion of 'close-copy stylization' between two inflection points. A complete phonetic description of the stylized f0 curve should capture its shape and its relation with the phone sequence. So it will classify its segments according to their size, direction (rise/fall) and position in the syllable. The principles for this classification are presented in 't Hart *et al.* (1990), which also provides a set of labels explicitly intended for Dutch. It should be noted that work proposes a methodology rather than a notational standard. There have been several applications of the IPO approach to different languages (such as English [Willems *et al.*, 1988], French [Beaugendre *et al.*, 1992], Italian [Quazza, 1991] or German [Brindopke *et al.*, 1997]) and different symbols have been used for the same concepts. Here, to keep to a concrete and classical example, we refer to the original proposal for Dutch.

In the IPO approach, pitch movements are intended to be superimposed on an ideal declination grid, which determines the height of the flat movements: two main declination lines (at least for Dutch) are identified as trends in the sequence of peaks and valleys. Pitch movements can follow the baseline or the topline, or depart from them. Every pitch movement departing from the declination lines can be characterized in terms of the following parameters:

a) direction (rise/fall)

b) timing (early in the syllable/late/very late)

c) rate of change (fast/slow)

d) size (full/half)

The combination of these features provides a set of possible pitch movements, which are labelled with a figure (if the movement is rising) or a letter (if the movement is falling):

| transcription symbol | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | A | B | C | D | E |
| **Direction** | | | | | | | | | | |
| rise | x | x | x | x | x | | | | | |
| fall | | | | | | x | x | x | x | x |
| **Timing** | | | | | | | | | | |
| early | x | | | | x | | x | | | x |
| late | | | x | | | x | | | | |
| very late | | x | | | | | | x | | |
| **Rate of change** | | | | | | | | | | |
| fast | x | x | x | | x | x | x | x | | x |
| slow | | | | x | | | | | x | |
| **Size** | | | | | | | | | | |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| full | x | x | x | x | | x | x | x | x | |
| half | | | | | x | | | | | x |

'Flat' pitch movements following the baseline or the topline are labelled with 0 or Ø respectively. A special diacritic '&' is used in the IPO transcription to join pitch movements occurring on the same syllable. For example a rise-fall with the peak in the middle of the syllable (pointed hat) could be labeled "1&A". In our formalization, where labels are assigned to <pitmove> elements, the diacritic '&' before a label will have the meaning "pitch movement realized on the same syllable as the preceding one". So, for example, a complex configuration rise-fall-rise occurring on a single syllable could be represented by three <pitmove>'s respectively labeled "1" "&A" "&2". Of course, two 'early' movements can't occur on the same syllable, so labels 1, 5, B, E can't be preceded by '&'.

#### 5.2.3.2 Data Source

The IPO notation scheme annotates pitch movements taking as a starting point the ‘close copy’ stylisation. In order to select the proper labels also phonetic segmentation is a necessary reference, allowing to identify syllables.

#### 5.2.3.3 Segmentation/selection

The IPO phonetic representation of intonation is a segmentation of the speech flow into consecutive pitch movements. Each <pitmove> covers a segment of the close copy stylized curve, the stretch between a <closecopy> inflection point and the following one. The annotator will label pitch movements by looking at the <closecopy>'s sequence, graphically displayed and aligned with the phonetic transcription of the utterance. He will define a <pitmove> by selecting two consecutive <closecopy> elements, from which the <pitmove> will inherit its time attributes start, end. Then, on the basis of the shape of the segment as displayed in the stylized curve and of its alignment with phones (syllable), he will assign the <pitmove> a proper label.

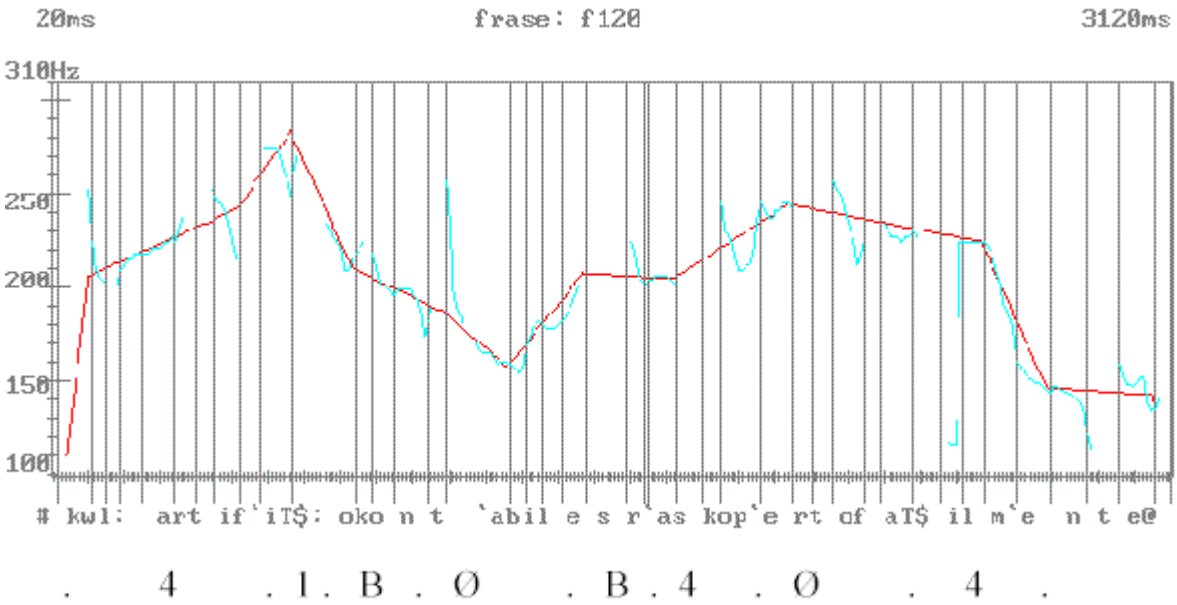
#### 5.2.3.4 Assignment

The attributes considered here for the <pitmove> element are the following:

- **type**: IPO symbol representing the movement
- **href**: two consecutive <closecopy> elements initial and final inflection points of the movement; alternatively, the <pitmove> might be linked with a <syllable>
- **start**: time start of the movement (inherited from the first linked <closecopy>)
- **end**: time end of the movement (inherited from the second linked <closecopy>)

### 5.2.4 Example

The following example shows the Italian sentence *"quell'artificio contabile sara' scoperto facilmente*" read by a female speaker. In the picture, the vertical bars correspond to phoneme boundaries (phoneme symbols are not SAMPA...), the blue line to the original f0 curve and the red line to the stylized one (closecopy.




This example can be represented using the <closecopy> and <pitmove> elements as below. It is assumed that the closecopy is directly imported as a sequence of inflection points (in this case f0.xml is not needed).

closecopy.xml

```
<closecopy id="clscpy_001" value="207" start="130"  end="130"/>
<closecopy id="clscpy_002" value="243" start="540"  end="540"/>
<closecopy id="clscpy_003" value="285" start="690"  end="690"/>
<closecopy id="clscpy_004" value="212" start="860"  end="860"/>
<closecopy id="clscpy_005" value="189" start="1110" end="1110"/>
<closecopy id="clscpy_006" value="159" start="1290" end="1290"/>
<closecopy id="clscpy_007" value="209" start="1500" end="1500"/>
<closecopy id="clscpy_008" value="206" start="1750" end="1750"/>
<closecopy id="clscpy_009" value="246" start="2070" end="2070"/>
<closecopy id="clscpy_010" value="226" start="2600" end="2600"/>
<closecopy id="clscpy_011" value="148" start="2780" end="2780"/>
<closecopy id="clscpy_012" value="144" start="3070" end="3070"/>
```

pitmove.xml

```
<pitmove id="pitm_001" type="4" href="closecopy.xml# id(clscpy_001).. id(clscpy_002)"/>
<pitmove id="pitm_001" type="1" href="closecopy.xml# id(clscpy_002).. id(clscpy_003)"/>
<pitmove id="pitm_001" type="B" href="closecopy.xml# id(clscpy_003).. id(clscpy_004)"/>
<pitmove id="pitm_001" type="Ø" href="closecopy.xml# id(clscpy_004).. id(clscpy_005)"/>
<pitmove id="pitm_001" type="B" href="closecopy.xml# id(clscpy_005).. id(clscpy_006)"/>
<pitmove id="pitm_001" type="4" href="closecopy.xml# id(clscpy_006).. id(clscpy_007)"/>
<pitmove id="pitm_001" type="Ø" href="closecopy.xml# id(clscpy_007).. id(clscpy_008)"/>
<pitmove id="pitm_001" type="4" href="closecopy.xml# id(clscpy_008).. id(clscpy_009)"/>
<pitmove id="pitm_001" type="0" href="closecopy.xml# id(clscpy_009).. id(clscpy_010)"/>
<pitmove id="pitm_001" type="B" href="closecopy.xml# id(clscpy_010).. id(clscpy_011)"/>
<pitmove id="pitm_001" type="Ø" href="closecopy.xml# id(clscpy_011).. id(clscpy_012)"/>
```

### 5.2.5 Coding Procedure

The objective of phonetic transcription of intonation according to the IPO methodology is to obtain a stylized curve where the sequence of pitch movements is properly labeled. The MATE workbench will not provide a true stylization/resynthesis environment. It might provide a pitch

tracking function to obtain the raw f0 curve, or alternatively a means to import it from external files. The most IPO-conformant coding procedure will directly import the stylized f0 curve, obtained with the help of a proper external environment for perceptual stylization (e.g. Winpitch, see http://www.winpitch.com), using the <closecopy> element with no need of the <f0> element, and will consist in the following steps:

- open the speech file in order to listen to its intonation
- open the corresponding phonetic segmentation (`<phone>` and `<syllable>`)
- import the close copy and display it as a curve, aligned with phonetic segmentation
- define `<pitmove>` elements by selecting the segments of the stylized curve (delimited by two consecutive `<closecopy>` elements) and labeling each of them according to the following criteria:
  - if it can be considered to coincide with the ideal baseline or topline, by a global look at the curve, label it 0 or Ø respectively
  - otherwise choose the proper label on the basis of movement direction and size and of its position in the syllable, judged by looking at its phonetic alignment

If the close copy is not available, the third step may be replaced by the following steps (a very simplified approximation of the correct stylization procedure):

- import or generate automatically the raw f0 curve and display it
- obtain a closecopy by selecting the 'relevant' `<f0>` points on the raw curve; base such stylization on the shape of the curve, the perceived intonation of the sound file and the alignment with syllables (accents, boundaries...)

### 5.2.6 Markup Table

| <closecopy> | |
|---|---|
| id | [ASCII] |
| value | [FLOAT] |
| href | <f0> |
| start | [FLOAT] |
| end | [FLOAT] |

| <pitmove> | |
|---|---|
| id | [ASCII] |

| | |
|---|---|
| type | 0, Ø, 1, 2, 3, 4, 5, A, B, C, D, E, &2, &3, &4, &A, &C, &D |
| href | <closecopy>.. <closecopy> or <syllable> |
| start | [FLOAT] |
| end | [FLOAT] |

## 5.3. Layer 2: Phonetic Representation of Intonation - INTSINT scheme

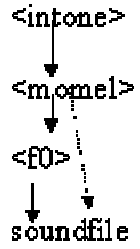
### 5.3.1 Markup Declaration

INTSINT is a coding system of intonation developed by Daniel Hirst and his colleagues at the CNRS centre of theAix-en-Provence University. It is conceived "to provide a purely formal encoding of the macroprosodic curve. Each target point of the stylised curve is coded by a symbol either as an absolute tone, defined globally with respect to the speakers pitch-range or as a relative tone, defined locally with respect to the inmediately neighbouring target-points"(Campione *et al.*, 1997, p. 72). Descriptions of this method can be found in Hirst (1991,1994); Hirst & Di Cristo (1998), among other references.

The starting point is again the raw f0 curve, which is (automatically) stylized to remove irrelevant and micro-prosodic details. The stylized representation, called MOMEL (Hirst & Espesser, 1993), consists in a sequence of inflection points [time : f0 value], which should be interpolated by a parabolic function. As a second step, each target point in the MOMEL stylized curve is considered in its absolute or relative height and accordingly labeled as a high or low tone. The elements necessary to represent the INTSINT notation system are the following:

- `<f0>`, for the frames of the raw f0 curve
- `<momel>`, for the inflection points in the stylized curve
- `<intone>`, for the labeled tones

The three elements are hierarchically ordered, with a one-to-one mapping between `<intone>`'s and `<momel>`'s. The alignment with the soundfile is kept through the base element `<f0>`, although in case the `<momel>` stylized curve is directly imported, the link with `<f0>` can be skipped and `<momel>` can be directly aligned with the soundfile.




### 5.3.2 The `<momel>` element

#### 5.3.2.1 Description

This element has been included to allow each f0 inflection point of a MOMEL stylised f0 contour (used in the INTSINT annotation system as the phonetic base representation of f0 contour) to be considered –and accordingly handled– as an XML element.

For a detailed description of the MOMEL stylization procedure, the reader is referred to Hirst & Espesser (1993), and Hirst (1994), among other references.

#### 5.3.2.2 Data Source

The MOMEL stylised f0 contour is obtained automatically from the raw f0 curve.

#### 5.3.2.3 Segmentation/selection

The calculated MOMEL stylised f0 values (or imported from the 'mes' tool) will be automatically converted to `<momel>` elements. 'Mes' is described at (and can be downloaded from) the following site: 'http://www.lpl.univ-aix.fr/ext/projects/mes_signaix.htm/'.

#### 5.3.2.4 Assignment

The attributes considered here for the <momel> elements are the following:

- **value**: f0 value (in Hz) of the stylised point
- **href**: `<f0>`, optional
- **start:** time start of the stylised point
- **end**: time end of the stylised point

If the MOMEL curve is imported from the 'mes' tool, the reference to `<f0>` can be avoided. In this case, as each inflection point is indeed a point, the two time attributes will have the same value. Otherwise, they will be inherited from start, end of the `<f0>` frame.

### 5.3.3 The `<intone>` element

#### 5.3.3.1 Description

The target points in the MOMEL stylized curve can be phonetically labeled as tones, here represented by the `<intone>` element.

INTSINT includes two types of symbols to transcribe f0 tones:

1) Absolute Tones

INTSINT includes three symbols to label the Absolute Tones, which are defined according to the speaker's pitch range.

| | |
|---|---|
| T | top of the speaker's pitch range |
| M | initial, mid value |
| B | bottom of the speaker's pitch range |

2) Relative tones

Relative tones are coded in INTSINT considering the height of the preceding and following target points. Five different symbols exist to transcribe these Relative Tones:

| | |
|---|---|
| H | target higher than both immediate neighbours |
| L | target lower than both immediate neighbours |
| S | target not different to preceding target |
| U | target in a rising sequence |
| D | target in a falling sequence |

#### 5.3.3.2 Data Source

The INTSINT representation is usually obtained from the MOMEL stylised f0 contour. So the `<intone>` element will be directly linked to `<momel>`. Phonetic segmentation is also useful to assign labels, although it is not strictly necessary.

#### 5.3.3.3 Segmentation/selection

The INTSINT symbols are assigned to each inflection point of the MOMEL stylised contour, following a set of conventions which are described in Hirst (1991, 1994), Hirst *et al.* (1993) and Hirst & Di Cristo (1998), among other references.

In order to label `<intone>`'s, the `<momel>` elements should be displayed as a stylized curve (parabolic interpolation) aligned with phonetic segmentation.

The INTSINT symbols can also be automatically assigned to the MOMEL inflection points by means of the 'mes' tool.

#### 5.3.3.4 Assignment

The attributes considered here for the <intone> element are the following:

- **type**: the INTSINT symbol corresponding to the tone.
- **href**: points to a single `<momel>` element
- **start**: time start of the stylised point, inherited from `<momel>`

- **end**: time end of the stylised point, inherited from <momel>

### 5.3.4 Example

The example presented here shows the MOMEL and INTSINT annotation of the French utterance *'Il faut que je sois a Grenoble Samedi vers quinze heures'*, using the <momel> and <intone> elements.

momel.xml

```
<momel id="mml_001" value="163" start="106"  end="106"/>
<momel id="mml_002" value="217" start="265"  end="265"/>
<momel id="mml_003" value="148" start="521"  end="521"/>
<momel id="mml_004" value="190" start="617"  end="617"/>
<momel id="mml_005" value="130" start="827"  end="827"/>
<momel id="mml_006" value="223" start="1249" end="1249"/>
<momel id="mml_007" value="139" start="1614" end="1614"/>
<momel id="mml_008" value="172" start="1822" end="1822"/>
<momel id="mml_009" value="144" start="1983" end="1983"/>
<momel id="mml_010" value="185" start="2078" end="2078"/>
<momel id="mml_011" value="152" start="2248" end="2248"/>
<momel id="mml_012" value="99"  start="2505" end="2505"/>
<momel id="mml_013" value="152" start="2730" end="2730"/>
```

intone.xml

```
<intone id="intn_001" type="L" href="momel.xml#id(mml_001)"/>
<intone id="intn_002" type="T" href="momel.xml#id(mml_002)"/>
<intone id="intn_003" type="M" href="momel.xml#id(mml_003)"/>
<intone id="intn_004" type="H" href="momel.xml#id(mml_004)"/>
<intone id="intn_005" type="L" href="momel.xml#id(mml_005)"/>
<intone id="intn_006" type="T" href="momel.xml#id(mml_006)"/>
<intone id="intn_007" type="M" href="momel.xml#id(mml_007)"/>
<intone id="intn_008" type="H" href="momel.xml#id(mml_008)"/>
<intone id="intn_009" type="L" href="momel.xml#id(mml_009)"/>
<intone id="intn_010" type="H" href="momel.xml#id(mml_010)"/>
<intone id="intn_011" type="D" href="momel.xml#id(mml_011)"/>
<intone id="intn_012" type="B" href="momel.xml#id(mml_012)"/>
<intone id="intn_013" type="M" href="momel.xml#id(mml_013)"/>
```

### 5.3.5 Coding Procedure

A specific tool, 'mes' (available at 'http://www.lpl.univ-aix.fr/ext/projects/mes_signaix.htm/'), has been developed to perform automatic intonation transcription according to the INTSINT system. Both stylization and annotation can be performed automatically by 'mes'. So, the simplest way to get to INTSINT annotation in the MATE environment would be the following:

- import the raw f0 curve in <f0>
- import the MOMEL stylized curve in <momel>
- import the INTSINT annotation in <intone>
- link <momel> to <f0> and <intone> to <momel> (an automatic function should be provided for that by the workbench)

The <f0> element may not be necessary (unless it is used as a reference by other layers...). In case only <momel> is imported, <intone>'s may be created manually by the following procedure:

- open the speech file in order to listen to its intonation

- open the corresponding phonetic segmentation
- import <momel> elements and display them as a stylized curve
- define <intone> elements by selecting every <momel> element (inflection point in the stylized curve) and mark it with the proper label

### 5.3.6 Markup Table

| <momel> | |
|---|---|
| id | [ASCII] |
| value | [FLOAT] |
| href | <f0> (optional) |
| start | [FLOAT] |
| end | [FLOAT] |

| <intone> | |
|---|---|
| id | [ASCII] |
| type | T, M, B, H, S, L, U, D |
| href | <momel> |
| start | [FLOAT] |
| end | [FLOAT] |

# 6. Layer 3: Phonological Representation of Intonation - ToBI scheme

## 6.1 Markup Declaration

Within the ToBI system [Silverman *et al.*, 1992], the Tone Tier is the level used to transcribe intonation phenomena. The types of phenomena covered by ToBI are tones and dowstepping, in their definition by Pierrehumbert (1980). F0 range and peak delay are also considered. The system is mainly phonological, labeling intonation events according to their function as described in language-dependent intonation models, with explicit reference to prosodic units such as prominent syllables, words and prosodic phrases. Nevertheless, some direct reference to acoustic values is admitted: while intonation events are supposed to be associated with linguistic units (syllables, words) they may also be aligned to specific points in the raw f0 curve, possibly corresponding to the tone 'peak'. Such alignment may be given for each tone or, alternatively, only for those whose peak occur too 'early' or too 'late' with respect to the stressed syllable. Moreover, a special marker may indicate the highest point in the f0 curve, to give an idea of the

pitch range. These acoustic references look spurious in phonological annotation and are required only when a true acoustic-phonetic representation of the curve is lacking. In the MATE meta-scheme, such intermediate level is present, so that it could be profitably used instead of the direct references to f0 points.

In our XML adaptation of ToBI, four elements have been defined:

- `<tobitone>`, for the *tones*, distinguished according to their function as *pitch accents*, *phrase accents* or *boundary tones* and labeled according to a classification of their linguistically admissible types
- `<target>`, to mark peak location when it occurs outside the scope of the accented syllable
- `<f0range>`, to mark the highest f0 value in the curve
- `<repair>`, to mark the restart of the intonation contour after a disfluency

The four elements are not hierarchically ordered. All *may* refer to the f0 curve, while only the two accessory element `<target>` and `<f0range>` are *necessarily* linked to `<f0>`. The `<tobitone>` and `<repair>` elements can be linked to prosodic units and/or to phonetic descriptions of intonation, rather than raw f0. This kind of reference is recommended. The other two elements are provided for completeness with respect to ToBI, but may be avoided.

## 6.2 The `<tobitone>` element

### 6.2.1 Description

The `<tobitone>` element has been defined to adapt to XML format the ToBI labels defined in the Tone Tier for the description of tones. In this framework, a tone is a functionally simple prosodic event which may be phonetically complex, e.g it may consist in an accent realized by reaching a low target f0 and immediately rising to a high f0 value. In fact, while the base descriptive elements are *pitch levels* H (high) and L (low), a tone can in some cases be described by a combination of levels, which amounts to describing it as a *movement*. ToBI notation presupposes a classification of the different types of tones admissible in a given language, so it is model and language dependent. In the following, the reference for the inventory of tones is the original ToBI model proposed for American English (Beckman & Ayers, 1994).

Within the ToBI framework, two types of tones are considered:

a) phrasal tones: pitch events associated with intonational boundaries.

b) pitch accents: pitch events associated with accented syllables.

Phrasal tones could be further distinguished into:

a.a) phrase accents, events at intermediate phrase boundaries

a.b) boundary tones, events at full intonation phrase boundaries

Note that, in the prosodic structure, an intonation phrase is a sequence of intermediate phrases, so it will be marked both by the phrase accent of its last intermediate phrase and by the boundary tone.

In our ToBI adaptation, we will use the `<tobitone>` element for all these classes of tones and distinguish them with the attribute *class*, which may assume the values *pitch accent* ("pitaccent"), *phrase accent* ("phraccent") or *boundary tone* ("boundtone"). For each class, a set of different tone types is defined, represented as values of the *type* attribute.

For a more detailed description of these symbols, the user is referred to Price (1992), Silverman *et al.* (1992), Beckman & Ayers (1994) and Pitrelli *et al.* (1994).

#### 6.2.1.1 Pitch accents

The inventory of pitch accents considered in ToBI is the following:

| | |
|---|---|
| H* | 'peak accent', an apparent tone target on the accented syllable which is in the upper part of the speaker's pitch range for the phrase. |
| L* | 'low accent', an apparent tone target on the accented syllable which is in the lowest part of the speaker's pitch range |
| L*+H | 'scooped accent', a low tone target on the accented syllable which is immediatly followed by relatively sharp rise to a peak in the upper part of the speaker's pitch range |
| L+H* | 'rising peak accent', a high peak target on the accented syllable which is immediatly preceded by relatively sharp rise from a valley in the lowest part of the speaker's pitch range |
| H+!H* | a clear step down onto the accented syllable from a high pitch which itself cannot be accounted for by a H phrasal tone ending the preceding phrase or by a preceding H pitch accent in the same phrase |

#### 6.2.1.2 Phrase accents

The different types of phrase accents considered in ToBI are:

| | |
|---|---|
| L- | Low phrase accent, which occurs at an intermediate phrase boundary |
| H- | High phrase accent, which occurs at an intermediate phrase boundary |
| !H- | Downstepped high phrase accent |

#### 6.2.1.3 Boundary tones

The different types of boundary tones are:

| | |
|---|---|
| L% | Low (final) boundary tone, which occurs at every full intonation phrase boundary |
| H% | High (final) boundary tone, which occurs at every full intonation phrase boundary |
| %H | Initial boundary tone; marks a phrase that begins relatively high in the speaker's pitch range; the default initial boundary is in the middle of the range or lower, and will be left unmarked in the transcription |

### 6.2.1.4 Uncertainty

ToBI has a way of dealing with uncertainty, by using one or several of the following symbols, that we will consider as special cases of <tobitone> elements:

| | |
|---|---|
| * | The pitch accent has not been transcribed yet |
| *? | Uncertainty about the presence of a pitch accent |
| X*? | Uncertainty about the type of pitch accent |
| - | The phrase accent has not been transcribed yet |
| -? | Uncertainty about the presence of a phrase accent |
| X-? | Uncertainty about the type of phrase accent |
| % | The boundary tone has not been transcribed yet |
| %? | Uncertainty about the presence of a boundary tone |
| X%? | Uncertainty about the type of boundary tone |

### 6.2.1.5 'Phrase accent' + 'boundary tone' combinations

As full intonation phrase boundaries will always have two final tones, a phrase accent tone plus a boundary tone, the possible set of allowable combinations at the end of an intonation unit is the following:

| | |
|---|---|
| L-L% | a full intonation phrase with an L phrase accent ending its final intermediate phrase and an L% boundary tone falling to a point low in the speaker's range<br>(standard 'declarative' contour of American English). |

| | |
|---|---|
| L-H% | a full intonation phrase with an L phrase accent closing the last intermediate phrase, followed by an H boundary tone (‘continuation rise’) |
| H-H% | an intonation phrase with a final intermediate phrase ending in an H phrase accent and a subsequent H boundary tone (‘yes-no questions’) |
| H-L% | an intonation phrase in which the H phrase accent of the final intermediate phrase upsteps the L% to a value in the middle of the speaker’s range (final level ‘plateau’) |

### 6.2.2 Data Source

A ToBI transcription is usually carried out taking the raw f0 as basic representation. Alternatively, it can rely on a phonetic representation of the f0 curve. In any case it should be aligned with linguistic units: phones or syllables or words or phrases or all of them.

### 6.2.3 Segmentation/selection

Tones are identified by inspecting the intonation curve (raw or stylized) and the aligned syllables and prosodic phrases. Depending on the annotation purposes, tones may be linked to points in the f0 curve or to linguistic units. ToBI annotation is not intended as a segmentation of the intonation curve, rather it is a selection of its relevant events, driven by the underlying linguistic structure of the utterance. ToBI tones are originally intended as associated to syllables, stressed syllables or phrase-final syllables, with some loose suggestion as to their precise alignement with the f0 curve: the ’peak’ or ’valley’ of the tone is intended to occur in the scope of the associated syllable, unless otherwise specified by the symbols ">" and "<" (see the `<target>` element in par. 6.3.1). In current implementations of the system, such as the one in the ESPS-Waves+ environment (‘http://www.ling.ohio-state.edu/phonetics/ToBI/ToBI.0.html’, ‘http://www.entropic.com/products&services/esps/esps.html’), the link with the f0 curve is made explicit, with tones aligned with their target point in the f0 curve.

Different options are available in MATE for `<tobitone>`'s alignment. A very simple one could be to have a display of the f0 raw curve aligned with `<word>`'s and define each `<tobitone>` by selecting the word on which it occurs. One could proceed in a similar way with `<syllable>`'s instead of words and select the syllable (or the `<phone>` sequence) to be associated with the tone.

In a more sophisticated approach, one may build up `<tobitone>`'s starting from the stylized curve, thus giving a phonetic correlate to the phonological ToBI label. To this end one may rely on `<pitmove>`'s or `<intone>`'s. The latter are perhaps more suitable to be considered as components of a `<tobitone>`, because of their underlying *pitch level* or *target point* interpretation. Basing on the synchronized display of the `<intone>` stylized curve, the `<phone>` transcription and the prosodic phrasing (`<breakindex>`), a `<tobitone>` will be defined by selecting its target points on the stylized curve: e.g. a simple `<tobitone>` like H* will be associated with a single `<intone>`, a complex one with a sequence of `<intone>`'s. Time attributes will be inherited from the selected `<intone>`'s. Time alignment will always be available in order to find out (via query or window synchronization) the corresponding `<syllable>`.

If an explicit association with f0 values is needed, one could code the f0 curve at level 2 in the <f0> element, and then associate <tobitone>'s with <f0> elements, or, alternatively, keep the link with <syllable> (or <word>) but set the time attribute to a time value corresponding to the f0 peak (that will then be retrievable via query).

### 6.2.4 Assignment

The attributes considered here for the <tobitone> element are the following:

- **type**: one of the labels defined in ToBI for tonal transcription.
- **class**: one of the class of tones defined in the ToBI system: "pitaccent" (pitch accent), "phraccent" (phrase accent), "boundtone" (boundary tone).
- **href**: <f0> or <intsint> or <syllable> or <word>
- **start**: time start of the tone (inherited)
- **end**: time end of the tone (inherited)

## 6.3 The <target> element

### 6.3.1 Description

This element is used to indicate the location of the f0 peak or valley of a pitch accent, when it does not coincide with the stressed syllable. In the original ToBI notation, such early or late target points are marked with the symbols ">" and "<", respectively.

### 6.3.2 Data Source

The raw or stylized f0 contour (in one of the following representations: <f0>, <closecopy>, <momel>, <intone>).

### 6.3.3 Segmentation/selection

The target position is located by visual inspection of the f0 contour. The corresponding <f0> (or stylized) element is selected and marked as 'EarlyF0', if it precedes the stressed syllable, or 'LateF0', if it follows it.

### 6.3.4 Assignment

The attributes considered here for the <target> element are the following:

- **type**: "EarlyF0" or "LateF0"
- **href**: <f0> or <closecopy> or <momel> or <intone>

- **start**: time start of the f0 peak (inherited)
- **end**: time end of the f0 peak (inherited)

## 6.4 The `<f0range>` element

### 6.4.1 Description

This element has been included to represent the ‘f0 range’ annotation symbol, which is used to indicate the f0 maximum in the speaker’s range for a given phrase.

### 6.4.2 Data Source

The raw or stylized f0 contour (in one of the following representations: `<f0>`, `<closecopy>`, `<momel>`, `<intone>`).

### 6.4.3 Segmentation/selection

The location of the maximum of the f0 range position is determined by visual inspection of the f0 contour and the corresponding element (`<f0>` or `<momel>` or `<closecopy>` or `<intone>`) is selected to be associated with the `<f0range>` element.

### 6.4.4 Assignment

The attributes considered here for the `<target>` element are the following:

- **type**: ToBI symbol for f0 maximum: "HiF0"
- **href**: `<f0>` or `<closecopy>` or `<momel>` or `<intone>`
- **start**: time start of the f0 peak (inherited)
- **end**: time end of the f0 peak (inherited)

## 6.5 The `<repair>` element

### 6.5.1 Description

This element has been included to represent the ‘repair’ annotation symbol "%r", defined in ToBI for the restart of an intonation contour when the last contour was interrupted without being finished by some disfluency. Such restart can be considered an intonation event aligned with a specific point in the f0 curve or with the corresponding prosodic unit.

### 6.5.2 Data Source

The raw or stylized f0 contour (in one of the following representations: <f0>, <closecopy>, <momel>, <intone>) and the phonetic transcription, with <phone> and <syllable> elements.

### 6.5.3 Segmentation/selection

Both listening and inspection of the f0 curve are necessary, aligned with phonetic transcription.

As in the case of <tobitone>, two main options are available: 1) select the <syllable> element on which the intonation restart occurs, 2) select an element in a phonetic representation of intonation: <f0>, <intone>, etc.

### 6.5.4 Assignment

The attributes considered here for the <repair> attribute are the following:

- **type**: ToBI symbol for repair ("%r")
- **href**: <f0> or <closecopy> or <momel> or <intone> or <syllable>
- **start**: time start of the f0 peak (inherited)
- **end**: time end of the f0 peak (inherited)

## 6.5 Example

The following example shows the ToBI annotation of the English utterance "Show me the cheapest fare from Philadelphia to Dallas excluding restriction VU slash one" (obtained from the TOBI-TRAINING material), using the elements <tobitone> and <repair>. Note that in this case tones are linked (by means of the 'href' attribute) to <word> elements, in addition to time values.

tobitone.xml

```
<tobitone id="tbtn_001" type="H*"   class="pitaccent" href="word.xml#id(wrd_001)"
                                                      start="2052" end="2052"/>
<tobitone id="tbtn_002" type="L+H*" class="pitaccent" href="word.xml#id(wrd_004)"
                                                      start="2579" end="2579"/>
<tobitone id="tbtn_003" type="!H*"  class="pitaccent" href="word.xml#id(wrd_005)"
                                                      start="3065" end="3065"/>
<tobitone id="tbtn_004" type="L-"   class="phraccent" href="word.xml#id(wrd_005)"
                                                      start="3315" end="3315"/>
<tobitone id="tbtn_005" type="L%"   class="boundtone" href="word.xml#id(wrd_005)"
                                                      start="3315" end="3315"/>
<tobitone id="tbtn_006" type="L+H*" class="pitaccent" href="word.xml#id(wrd_009)"
                                                      start="4470" end="4470"/>
<tobitone id="tbtn_007" type="!H*"  class="pitaccent" href="word.xml#id(wrd_009)"
                                                      start="4771" end="4771"/>
<tobitone id="tbtn_008" type="L-"   class="phraccent" href="word.xml#id(wrd_009)"
                                                      start="5015" end="5015"/>
<tobitone id="tbtn_009" type="H*"   class="pitaccent" href="word.xml#id(wrd_011)"
                                                      start="5388" end="5388"/>
<tobitone id="tbtn_010" type="L-"   class="phraccent" href="word.xml#id(wrd_011)"
                                                      start="5855" end="5855"/>
<tobitone id="tbtn_011" type="L%"   class="boundtone" href="word.xml#id(wrd_011)"
                                                      start="5855" end="5855"/>
<tobitone id="tbtn_012" type="L+H*" class="pitaccent" href="word.xml#id(wrd_012)"
                                                      start="6984" end="6984"/>
<tobitone id="tbtn_013" type="L-"   class="phraccent" href="word.xml#id(wrd_012)"
```

```
                                                        start="7399" end="7399"/>
<tobitone id="tbtn_014" type="L%"   class="boundtone" href="word.xml#id(wrd_012)"
                                                        start="7399" end="7399"/>
<tobitone id="tbtn_015" type="H*"   class="pitaccent" href="word.xml#id(wrd_013)"
                                                        start="8154" start="8154"/>
<tobitone id="tbtn_016" type="L-"   class="phraccent" href="word.xml#id(wrd_013)"
                                                        start="8585" end="8585"/>
<tobitone id="tbtn_017" type="L%"   class="boundtone" href="word.xml#id(wrd_013)"
                                                        start="8585" end="8585"/>
<tobitone id="tbtn_018" type="H*"   class="pitaccent" href="word.xml#id(wrd_014)"
                                                        start="8711" end="8711"/>
<tobitone id="tbtn_019" type="!H*"  class="pitaccent" href="word.xml#id(wrd_015)"
                                                        start="8928" end="8928"/>
<tobitone id="tbtn_020" type="L-"   class="phraccent" href="word.xml#id(wrd_015)"
                                                        start="9114" end="9114"/>
<tobitone id="tbtn_021" type="H*"   class="pitaccent" href="word.xml#id(wrd_016)"
                                                        start="9353" end="9353"/>
<tobitone id="tbtn_022" type="H*"   class="pitaccent" href="word.xml#id(wrd_017)"
                                                        start="9694" end="9694"/>
<tobitone id="tbtn_023" type="L-"   class="phraccent" href="word.xml#id(wrd_017)"
                                                        start="9880" end="9880"/>
<tobitone id="tbtn_024" type="L%"   class="boundtone" href="word.xml#id(wrd_017)"
                                                        start="9880" end="9880"/>
```

repair.xml

```
<repair id="rpr_001" type="%r" start="4149" end="4149"/>
```

## 6.6 Coding Procedure

Different procedures may be followed to obtain a ToBI annotation of intonation. As above mentioned, a simple procedure may look at the shape of the raw f0 curve and align <tones> to <words>. Alternatively, tones may be linked to stylized curves or linguistic units.

A recommended procedure, in line with the multilevel integrated MATE approach, is the following (where <intone>'s could be replaced with <closecopy>'s):

1) open the following synchronized windows: <intone> (with the <momel> graphical display of the stylized f0 curve), <phone>, <syllable>, <breakindex>

2) look for stressed syllables in the <syllable> sequence, listen to the signal and inspect the corresponding f0 contour to judge if it is a pitch accent

3) if it is, select its components (<intone> elements) and create a corresponding <tobitone> (that will inherit time attributes from <intone>'s); for each detected intonation event, select the <syllable> on which it occurs, create the corresponding (linked) <tobitone> or <repair> element and assign it the proper class attribute; time values will be inherited from <syllable> or set explicitely to be aligned with the f0 peak

4) label it with class = pitaccent and type = the appropriate ToBI label

5) (alternatively link the tone to the corresponding stressed <syllable>, or to the corresponding <word>)

6) look for phrase boundaries in the <breakindex> stream, listen to the signal and inspect the corresponding f0 contour to recognize the type of phrasal accent

7) if the <breakindex> value is 4 (i.e. it corresponds to a full intonation boundary) decompose the contour into phrase accent and boundary tone

8) select the <intone> elements componing the phrase accent and create a corresponding <tobitone> element (that will inherit time attributes from <intone>'s) with class= phrase accent and type= appropriate ToBI label

9) if there is a boundary tone (break index 4), select its <intone> element and create the corresponding <tobitone> with class= boundary tone and appropriate label

10) (alternatively link the tone to the corresponding <syllable>)

With the explicit link to the phonetic representation of f0, the <target> and <f0range> elements may be unnecessary. If desired, they may be introduced by selecting the corresponding <intone> element.

## 6.7 Markup Table

| <tobitone> | |
|---|---|
| id | [ASCII] |
| type | L-, H-, !H-, -, -?, X-?, L-L%, L-H%, H-H%, H-L%, %, %?, X%?, H*, !H*, L*, L*+H, L*+!H, L+H*, L+!H*, H+!H*, *, *?, X*? |
| class | pitaccent, phraccent, boundtone |
| href | <f0> or <closecopy> or <momel> or <intone> or <syllable> or <word> |
| start | [FLOAT] |
| end | [FLOAT] |

The set of symbols defined for the attribute 'type' includes the allowable combination of pitch accents, phrase accents, boundary tones and/or uncertainty symbols, as defined in the ToBI guidelines.

The value for the attribute 'type' should be consistent with the attribute 'class', according to the semantics of the different labels described in the tables in 6.2.1.

| <target> | |
|---|---|
| id | [ASCII] |

| type | EarlyF0, LateF0 |
|---|---|
| href | <f0> or<br><closecopy> or<br><momel> or<br><intone> |
| start | [FLOAT] |
| end | [FLOAT] |

| <f0range> | |
|---|---|
| id | [ASCII] |
| type | HiF0 |
| href | <f0> or<br><closecopy> or<br><momel> or<br><intone> |
| start | [FLOAT] |
| end | [FLOAT] |

| <repair> | |
|---|---|
| id | [ASCII] |
| type | %r |
| href | <f0> or<br><closecopy> or<br><momel> or<br><intone> or<br><syllable> |
| start | [FLOAT] |
| end | [FLOAT] |

# 7. Layer 4: Prosodic Phrasing - ToBI scheme

## 7.1 Markup Declaration

The Prosodic Phrasing Layer is intended to represent the prosodic structure of the utterance, at the levels above the word. In this sense it is complementary to the Phonetic Transcription Layer, where sub-word units such as phones and syllables are represented. The ToBI annotation system provides an effective way of marking prosodic structure, in its Break-Index Tier. The underlying theory builds up prosodic units in a hierarchy where clitics are joined to the following content word, word sequences form intermediate phrases and intermediate phrases form full intonation phrases. The end of each constituent is marked by a prosodic event whose importance is proportional to the boundary depth. ToBI provides a series of *break indexes* to rate the depth of

the boundary. A single element <breakindex> is here defined to represent word boundaries and rate them with the proper degree of disjuncture.

## 7.2 The `<breakindex>` element

### 7.2.1 Description

ToBI symbols for prosodic boundaries can be adapted to XML by using a single element called `<breakindex>`. The ToBI notation conventions for Break Index transcription include the following symbols:

| | |
|---|---|
| 0 | for cases of clear phonetic marks of clitic groups; e.g. the medial affricate in contractions of 'did you' or a flap as in'got it' |
| 1 | most phrase-medial word boundaries |
| 2 | a strong disjuncture marked by a pause or virtual pause, but with no tonal marks; i.e. a well-formed tune continues across the juncture<br>OR<br>a disjuncture that is weaker than expected at what is tonally a clear intermediate or full intonation phrase boundary |
| 3 | intermediate intonation phrase boundary; i.e. marked by a single phrase tone affecting the region from the last pitch accent to the boundary |
| 4 | full intonation phrase boundary; i.e. marked by a final boundary tone after the last phrase tone |

ToBI includes also, for the representation of disfluencies, the 'p' diacritic, which can be attached to a break index symbol, as in the following example:

| | |
|---|---|
| 3p | a hesitation pause or a pause-like prolongation where there is a phrase accent in the tone tier |

Uncertainty and underspecification can also be expressed by means of the '-' and '?' diacritics, and the 'X' symbol:

| | |
|---|---|
| 1- | uncertainty between '0' and '1' values |

| | |
|---|---|
| 2- | uncertainty between ‘1’ and ‘2’ values |
| 3- | uncertainty between‘2’ and ‘3’ values |

| | |
|---|---|
| 1p? | uncertainty about ‘1p’ |
| 2p? | uncertainty about ‘2p’ |
| 3p? | uncertainty about ‘3p’ |

| | |
|---|---|
| X | underspecification of a break index value |

The combination of symbols and diacritics provides the different possible string symbols which can be used in the notation of prosodic boundaries by using the ToBI scheme.

For a more detailed description of these symbols, the user is referred to Price (1992), Silverman *et al.* (1992), Beckman & Ayers (1994) and Pitrelli *et al.*(1994).

### 7.2.2 Data Source

Break Index symbols are usually located at the end of words (not at the beginning of the unit); for this reason, it is useful to have available during the transcription task the orthographic transcription, as well as the speech file (and possibly the phonetic representation of intonation).

### 7.2.3. Segmentation/selection

Break indexes are assigned to a word in order to classify the degree of juncture perceived with the following word. For such classification, the speech file, aligned with its orthographic transcription should be available and listened to. The display of the corresponding stylized curve may be helpful. The `<breakindex>` element will be defined by selecting a `<word>` element, listening to the corresponding signal portion, looking at the f0 curve and on these basis classifying the type of right boundary of the word.

### 7.2.4. Assignment

The attributes considered here for the `<breakindex>` element are the following:

- **type**: the ToBI break-index label symbol classifying the prosodic boundary
- **href**: `<word>`
- **start**: time location of the boundary (inherited from `<word>` start)
- **end**: time location of the boundary (inherited from `<word>` end)

## 7.3 Example

The following example shows the 'break index' annotation of the same utterance of section 6.5 ("Show me the cheapest fare from Philadelphia to Dallas excluding restriction VU slash one") using the <breakindex> element:

breakindex.xml

```
<breakindex id="brkndx_001" type="1"  href="word.xml#id(wrd_001)" start="2105" end="2105"/>
<breakindex id="brkndx_002" type="1"  href="word.xml#id(wrd_002)" start="2245" end="2245"/>
<breakindex id="brkndx_003" type="1"  href="word.xml#id(wrd_003)" start="2355" end="2355"/>
<breakindex id="brkndx_004" type="1"  href="word.xml#id(wrd_004)" start="2935" end="2935"/>
<breakindex id="brkndx_005" type="4"  href="word.xml#id(wrd_005)" start="3315" end="3315"/>
<breakindex id="brkndx_006" type="1"  href="word.xml#id(wrd_006)" start="3565" end="3565"/>
<breakindex id="brkndx_007" type="1p" href="word.xml#id(wrd_007)" start="3836" end="3836"/>
<breakindex id="brkndx_008" type="1"  href="word.xml#id(wrd_008)" start="4325" end="4325"/>
<breakindex id="brkndx_009" type="3"  href="word.xml#id(wrd_009)" start="5015" end="5015"/>
<breakindex id="brkndx_010" type="1"  href="word.xml#id(wrd_010)" start="5225" end="5225"/>
<breakindex id="brkndx_011" type="4"  href="word.xml#id(wrd_011)" start="5855" end="5855"/>
<breakindex id="brkndx_012" type="4"  href="word.xml#id(wrd_012)" start="7399" end="7399"/>
<breakindex id="brkndx_013" type="4"  href="word.xml#id(wrd_013)" start="8585" end="8585"/>
<breakindex id="brkndx_014" type="1"  href="word.xml#id(wrd_014)" start="8825" end="8825"/>
<breakindex id="brkndx_015" type="3"  href="word.xml#id(wrd_015)" start="9115" end="9115"/>
<breakindex id="brkndx_016" type="1"  href="word.xml#id(wrd_016)" start="9595" end="9595"/>
<breakindex id="brkndx_017" type="4"  href="word.xml#id(wrd_017)" start="9880" end="9880"/>
```

## 7.4 Coding Procedure

A procedure for break index annotation may be:

1) open the following synchronized windows: speech file, f0 curve (<f0> or <closecopy> or <momel>...), <word>

2) select a word to define a corresponding <breakindex> element

3) listen to a surrounding portion of the speech signal and look at the f0 curve, in order to classify the boundary depth and choose the proper label

## 7.5 Markup Table

| <breakindex> | |
|---|---|
| id | [ASCII] |
| type | 0, 1-, 1, 1p, 1p?, 2-, 2, 2p, 2p?, 3-, 3, 3p, 3p?, 4-, 4, X |
| href | <word> |
| start | [FLOAT] |
| end | [FLOAT] |

The attribute ‘type’ can only contain an allowable combination of ToBI symbols, as described in 7.2.1.

# References

BEAUGENDRE, F. *et al.* (1992).- "A perceptual study of French intonation*", ICSLP 92, Banff*.

BECKMAN, M.E. - AYERS, G.M. (1994).- *Guidelines for ToBI Labelling*. Version 2.0, February1994.

BECKMAN, M.E.- HIRSCHBERG, J. (1994).- *The ToBI AnnotationConventions*. Appendix A of BECKMAN, M.E. - AYERS, G.M.(1994).- *Guidelines for ToBI Labelling.*Version 2.0, February 1994.

BRINDOPKE, C. - PHADE, A. - KUMMERT, F. - SAGER, G. (1997).- "An envorinment for the labeling and testing of melodic aspects of speech", *Eurospeech 97*.

CAMPIONE, E. - FLACHAIRE, E. - HIRST, D. - VERONIS, J. (1997).- "Stylisation and symbolic coding of F0: A quantitative model", en BOTINIS, A. - KOUROUPETROGLOU, G. - CARAYIANNIS, G. (eds.) (1997).- *Intonation: Theory, Models and Applications. Proceedings of an ESCA Workshop, September 18-20, Athens, Greece*, ESCA/Universty of Athens, pp. 71-74.

GIBBON, D. (1989).- *Survey of Prosodic Labelling for EC Languages*. SAM-UBI-1/90, 12 February 1989; Report e.6, in ESPRIT 2589 (SAM) Interim Report, Year 1.=20, Ref.SAM-UCL G002. University College London, February 1990.

HIRST, D.J. (1991).- "Intonation models: Towards a third generation" in *Actes du XIIème Congrés International des Sciences Phonétiques. 19-24 août 1991, Aix-en-Provence, France. Aix-en-Provence*: Université de Provence, Service des Publications. Vol. 1 pp. 305-310.

HIRST, D.J. (1994).- "The symbolic coding of fundamental frequency curves: from acoustics to phonology", in FUJISAKI, H. (Ed.) *Proceedings of International Symposium on Prosody, Satellite Workshop ofICSLP 94, Yokohama, September 1994*.

HIRST, D.J. - DI CRISTO, A. - LE BESNERAIS, M. - NAJIM, Z. - NICOLAS, P. -ROMƒAS, P. (1993).- "Multilingual modelling of intonation patterns", in HOUSE, D.TOUATI, P. (Eds.) *Proceedings of an ESCA Workshop on Prosody. September 27-29, 1993, Lund,Sweden*. Lund University Department of Linguistics andPhonetics, Working Papers 41. pp. 204-207.

HIRST, D.J. - DI CRISTO, A. (Eds.) (1998).- *Intonation Systems. A Survey of Twenty Languages*. Cambridge:Cambridge University Press.

HIRST, D.J. - ESPESSER, R. (1993).- "Automatic modelling of fundamental frequency using a quadratic spline function", *Travaux de l‘Institut de Phonetique d’Aix*, 15, pp. 71-85.

PIERREHUMBERT, J. B. (1980).- *The Phonology and Phonetics of English Intonation*, Bloomington, Indiana UniversityLinguistics Club.

PITRELLI, J. - BECKMAN, M. - HIRSCHBERG, J. (1994).- "Evaluationof prosodic transcription labelling reliability in the tobiframework", en *Proceedings of the third International Conference on Spoken Language Processing*, Yokohama, ICSLP, Vol. 2, pp. 123-126.

PRICE, P. (1992).- *Summary of the Second Prosodic Transcription Workshop: the TOBI ( TOnes and Break Indices) Labeling System.* Nynex Science and Technology, Inc. 5-6 April, 1992. Linguist List vol.3-761, 9 October 1992.

QUAZZA, S. (1991).- "Modelling Italian Intonation on a Text-to-Speech System", *Eurospeech 91, Genova*.

SILVERMAN, K.- BECKMAN, M.- PITRELLI, J.-OSTENDORF, M.- WIGHTMAN, C.- PRICE, P.- PIERREHUMBERT, J.- HIRSCHBERG, J.(1992).- "TOBI: A standard for labeling English prosody", in OHALA, J.J. et al. (eds.).- *Proceedings ICSLP 92*, pp. 867-870.

'T HART, J. - COLLIER, R. - COHEN, A. (1990).- *A perceptual study of intonation. An experimental-phonetic approach to speech melody*, Cambridge, Cambridge University Press.

WELLS, J. BARRY, W. - GRICE, M. - FOURCIN, A. GIBBON, D. (1992).- *Standard Computer-Compatible Transcription*, SAM Stage Report Sen.3 SAM UCL-037, 28 February 1992. In SAM (1992)ESPRIT PROJECT 2589 (SAM) Multilingual Speech Input/Output Assessment,Methodology and Standardisation. *Final Report. YearThree*: 1.III.91-28.II.1992. Ref. SAM-UCL-G004. London: University CollegeLondon.

WILLEMS, N. J. - COLLIER, R. - T'HART, J. (1988).- "A synthesis scheme for British English intonation", *Journal of the Acoustical Society of America*, 84.

# MORPHOSYNTAX

**Vito Pirrelli, Claudia Soria**

# 1 Introduction

The present document is intended to offer an edited selection of *good practices* for annotation of dialogue text at the levels of word analysis, chunking and representation of syntactic functional relations (sections 3, 4 and 5). Moreover, it provides guidelines to the *edited* transcription of a dialogue (section 2). These levels are seen as conceptually independent, though inter-connected, sub-levels of morpho-syntactic analysis. For a more comprehensive review of the concrete practices followed in a representative set of current annotation schemes, the interested reader is referred to MATE Deliverable 1.1. Here, we provide a formal framework, or annotation *meta-scheme*, to be used as a practical blue-print to both language- and domain-specific scheme development and/or customization. Incidentally, it should be emphasized that the meta-scheme described in these pages offers the further bonus of being usable as a kind of *lingua franca* for exchange of information and data. This is supported by the choice of XML as mark-up language.

## 1.1 General Requirements

We first identified the following list of prerequisites to the design of a meta-scheme for dialogue annotation at the morpho-syntactic level:

- be robust and wide-coverage;
- be flexible, customizable and usable for practical applications;
- be modular (to allow for partial instantiations of the meta-scheme);
- be redundantly specified (to be able to accomodate alternative practices for the annotation of the same phenomenon);
- make provision for graded levels of abstraction from raw data;
- be amenable to (semi)automatic annotation;
- be reliable in terms of inter-annotator agreement;
- have the potential for multi-lingual application.

In what follows, we will comment on each of these desiderata and illustrate their relationship with morpho-syntactic annotation in MATE.

### 1.1.1 Coverage and Expected Feed-back

It is useful, at this stage, to make it clear what sort of input the present deliverable is intended to elicit. First, we provide a list of phenomena modelled through the suggested meta-scheme. The list exemplifies a representative range of phenomena crucially involved in the annotation of a dialogue at the levels of morpho-syntactic analysis touched on in this report, but it is not intended to give instructions for marking up an exhaustive list of language-specific facts. For example, we suggest to annotate derivatives through *immediate morpheme segmentation*, i.e. by signalling the most external affix only, as in ‘‘(derivation(al))”. However, we do not provide language-specific recommendations concerning problems of segmentation due to fusional phenomena or truncated

stems, as in *provide* →*provision* or *truncate* →*truncation*. First, it would simply be impossible to tackle problems at this level of detail for even a subset of the languages that MATE is interested in covering. Secondly, linguistic issues at this level of granularity depend too heavily on the theoretical commitment of the annotators and on the intended purposes of their annotation scheme. It is questionable to suggest a standard practice at this level.

Furthermore, the range of phenomena to be considered poses a considerable challenge to any attempt to adapt existing annotation practices, predominantly designed for annotating written texts, to the specific exigencies of dialogic data. The challenge has mainly to do with the noisy nature of spoken texts: namely, usage of non-standard forms, repetitions, false starts, anacolutha and incomplete phrases, etc. We envisage that most of this noise should preliminarily be annotated at a low-level of *edited transcription* as shown in section 2 of the present report. This stage is intended as a preliminary filter and plays the role of marking noisy or non standard material with no editing out. In fact, much of noise will receive a linguistic annotation at the level of chunking (section 4 of the present report), while being ignored at the level of functional annotation.

### 1.1.2 Core and Periphery

For all sub-levels of annotation considered here, the meta-scheme consists of two subsets of tags. The first subset, or *core scheme*, supplies basic means for annotating obligatory information. The second subset, or *periphery tag set*, serves the purpose of making provision for further linguistic annotation to be added on top of obligatory information, whenever this is required by the annotator. In its turn, the periphery tag set parts into two further subsets: a recommended set and an optional one. This makes the meta-scheme highly modular, and open to further augmentation, both in terms of more granular information and of further independent dimensions of analysis.

### 1.1.3 Redundancy

In designing the MATE meta-scheme for morpho-syntactic annotation, considerable care was taken to provide the potential user with a battery of open choices for encoding a particular range of phenomena, rather than give one solution only. For example, a morphological compound can either be represented as a whole morphological word, or as two (or more) independent words linked together through a compounding relationship.

### 1.1.4 Supported Annotation Schemes

We provide, at each sub-level of annotation, an indication of some currently available mark-up schemes that i) are compatible with our meta-scheme, and ii) will be supported by the MATE Workbench. These recommended practices are to be interpreted as a fair basis for testing the MATE Workbench. Our main concern was to support those annotation practices which both represent current standardization efforts and are proven to be portable and usable for practical applications. Two efforts, in particular, meet these requirements, namely the EAGLES and SPARKLE projects. EAGLES annotation scheme is the product of a joint European EC-funded standardisation effort carried out in the framework of the European Action Group for Language Engineering Standards.

The output of EAGLES (http://www.ilc.pi.cnr.it/EAGLES/home.html) has been particularly instrumental for the definition of a common set of morphosyntactic tags, with some integration and extension. The EC-funded project SPARKLE (Shallow Parsing and Knowledge Extraction for Language Engineering (http://www.ilc.pi.cnr.it/sparkle.html)) has developed a layered scheme of syntactic annotation encompassing three different levels of annotation: the chunk-based level,

the constituency-based level and the functional level. The scheme is designed so as to be flexible, multipurpose, and amenable to finite state techniques of local and robust parsing.

In fact, SPARKLE specification were actually used for intelligent cross-lingual text editing and translation filtering in multilingual information retrieval systems (Xerox, Sharp), and speech recognition (Daimler-Benz), with a steady improvement in performance.

The annotation meta-scheme presented in this report represents an XML instantiation of two of the three levels envisaged in SPARKLE: namely chunking and functional annotation. In Deliverable 1.1 we argued that these two levels provide room for partial parsing, underspecification and graded levels of abstraction from raw input text, thus fulfilling many of the desiderata connected with syntactic processing of typically noisy data such as dialogues.

### 1.1.5 Amenability to Automatic Annotation

The concern for amenability to (semi)automatic annotation is of paramount importance throughout the present report, and has influenced a number of choices. For example, the preference given to *chunking* and *functional annotation* over, e.g., phrase structure trees at the syntactic level is chiefly motivated by the *locality* of the analysis offered by these two schemes, a useful feature if one wants to prevent a local parsing failure from backfiring and causing the entire parse of an utterance to fail. This is particularly desirable with a view to manual correction of an automatically annotated output, since a reasonably reliable partial analysis is always better than no analysisat all.

### 1.1.6 Reliability

Inter-coder human reliability deals with the issue of replicability of a task by a human annotator. The task is defined by an annotation scheme and a corpus to be annotated accordingly. The common assumption is that each token phenomenon in the corpus should be given the same tag by several independent annotators. A meta-scheme, as an edited collection of different annotation schemes, can only i) provide figures of the reliability of each independently supported scheme, and ii) possibly shed light on and give reasons for the respective degrees of reliability of the tag sets considered.

### 1.1.7 Multilinguality

The selection of a scheme is also motivated on a multi--lingual basis. In fact, consideration of multilingual aspects was one of the main motivations in support of our choice of the SPARKLE annotation scheme as a starting point here (see sections 4 and 5 below), since SPARKLE provides annotated material in four different languages, namely English, French, German and Italian. However, in the present meta-scheme no attempt is made at covering any one language in particular as the supported schemes present slightly different annotation choices, depending on language-specific phenomena. For example, while, for some languages, the syntactic function of a phrase can be read off a traditional constituency-based parse tree, it is not clear how tenable the same claim is for relatively free phrase order languages such as German and Italian, which typically exhibit complex cases of *phrase scrambling* and *discontinuous phrases*. This argument justifies our choice of considering syntactic functions as primitive linguistic notions, rather than somewhat derivative of the configurational properties of a language.

## 1.2 Overview

The present document is structured into four sections, apart from this one. Each section covers a level of morpho-syntactic annotation, and their sequence ideally represents the procedure which should be followed by the potential user in annotating a dialogue at the morpho-syntactic level. The individual sub-levels are:

- the *edited transcription level*, whereby the transcript is annotated for all nonstandard forms, repetitions, false starts etc.
- the *morphological word-level*, whereby morphological words are identified and annotated for their morphological properties. Annotation at this level is required for annotation at higher levels of analysis.
- the *chunk-level*, where the document is annotated for its syntactic structure into chunks (see section 3)
- the *functional-level*, whereby functional dependencies between lexical heads are annotated.

All levels presuppose a transcription file marked up in XML for words. Levels c) and d) are independent of each other, but dependent on level b).

## 1.3 Document Index


- **Section 2: Edited Transcription Coding Module**
- **Section 3: Morphological Annotation Coding Module**
- **Section 4: Chunk-level Annotation Coding Module**
- **Section 5: Functional Annotation Coding Module**
- **References**
- **EAGLES recommended attributes/values**

# 2 Edited Transcription Coding Module

## 2.1 Coding Purpose

The result of applying the Edited Transcription Coding Module to the transcription file will be a new file where the transcription is marked up for some basic dysfluency phenomena. Identification of basic dysfluency phenomena is logically prior to the levels of linguistic annotation "proper" defined in sections 3, 4 and 5. We conceive this preliminary marking of a transcription file as a sort of "minimal editing". In this perspective, editing a transcript involves identification and annotation of the following phenomena:

- repetitions, retrace and repair sequences, false starts
- non standard forms, errors
- omissions
- interruptions and completions

## 2.2 Existing Schemes

Probably the most crucial area where grammatical standards developed for written language need to be extended for annotation of spontaneous spoken material is that of speech repairs. Many attempts have been made at a rigorous categorization of speech repairs (cf., among the others, Levelt, 1989; Howell and Young, 1990, 1991; Meeter et al., 1995). Some annotation systems incorporate such theoretical approaches: see, for example, the CHRISTINE annotation system (drawing on Levelt and Howell & Young) or the Switchboard annotation system. A very detailed typology of speech repairs is offered by CHILDES (MacWhinney, 1991, 1994).

It is generally recognized that such accounts are either too fine grained, or too coarse grained. For example, they may fail, in some cases, to provide information about the relationship of the sequence containing the repair to the larger context of the embedding utterance, or the relationship between the reparandum and the repairing text sequence. In other cases, they require annotation of fairly subtle distinctions, e.g. between different types of interrupted linguistic units, which are either very difficult to draw in practice, or relevant to other levels of linguistic annotation (e.g. prosody or chunking). In this section, we made an attempt at overcoming these problems by defining three levels of annotation of speech repairs: i) obligatory information, ii) recommended information and iii) optional information. This reflects a similar approach taken in EAGLES for morpho-syntactic annotation (see section 3).

At level i), we introduce information concerning what portions of dialogue are to be edited (e.g. replaced, integrated, standardized, continued etc.), with an indication of the target portions intended to replace, integrate, standardize and continue edited material. At level ii), we suggest possible ways of typing the relationship between the edited and the editing text material: e.g., if there is a repetition, a retracing, a reformulation etc. Finally, level iii) annotates the role played by the excerpts identified at level i) in the relationship annotated at level ii). It is important to appreciate that these levels of annotation can be overlaid in an incremental fashion, so that one can reasonably start annotating a dialogue at level i), to progressively augment this information through addition of levels ii) and iii).

## 2.3 Markup Declaration

The overall set of mark-up elements and their hierarchical relationship are as follows:

```
<seg>
<dys>
     <repair>
     <reparandum>
     <signal>
```

### 2.3.1 <seg>

#### 2.3.1.1 Description

A <seg> element is an all-purpose element intended to mark dysfluent portions of dialogue and their possible repairs (when present in context). In most cases, <seg> elements contain sequences of orthographic words to be edited. In some cases, <seg> elements can also mark whether the annotated portion is to be edited out, standardized, replaced or integrated in some way.

Annotation through <seg> elements makes it possible for a dysfluent portion to be edited out from the text, and thus be ignored at the morphological and syntactic levels. Use of <seg> elements, as well as any decisions about their coverage, is of course a matter of choice. To a certain extent, in fact, the present meta-scheme allows some degree of information about dysfluent material to be expressed at later stages of analysis. For instance, information that a whole phrase was actually interrupted can also be annotated at the level where phrases are identified and marked up (e.g. at the chunking level, if appropriate). Annotation at the *Edited transcription* level is recommended whenever a detailed analysis of dysfluency phenomena is pursued.

<seg> elements convey basic, obligatory information. Further refinements of this obligatory information are possible through use of recommended and optional elements, which refer to <seg> elements through inline href links (see below, sections 2.3.2 and 2.3.3).

#### 2.3.1.2 Data Source

Tagging of <seg> presupposes the markup of orthographic words.

#### 2.3.1.3 Segmentation/Selection

<seg> elements serve to mark those portions of a transcript that the annotator wishes to edit. For this purpose, it is useful to distinguish two basic types of dysfluent material: a) dysfluencies whose repair is already provided by a speaker within the text: this is the case of repetitions, retracings, false starts, completions, etc.; b) dysfluencies which are not repaired further down in the text, but get edited in isolation whenever felt necessary: this is the case of utterances which are left incomplete, or of non-standard words/expressions, for which an annotator may wish to suggest a target standard form. It is important to note that while dysfluencies of type a) require identification of two portions of text (and thus two actually independent segments), namely the *reparandum* and the *repairing* sequence, which are intended to be in a parallel distribution

relative to the annotated text, annotation of dysfluencies of type b) requires identification of the text portion to be corrected only. From this it follows that in an utterance such as *I wanted I wanted to invite Margie* the whole repeated phrase *I wanted* should be signalled as one segment; the same holds for the ensuing phrase *I wanted*. In dysfluencies of type b), such as an utterance like *I didn't wan…*, where the speaker gets interrupted and is no longer given a chance of completing his/her utterance within the turn, we suggest to mark one segment only.

Different types of dysfluency require different criteria for segmentation of the relevant portions of text. In what follows, we provide a non exhaustive list of relevant phenomena, together with an illustration of the recommended segmentation strategy for each case.

**Nonstandard forms**

Spoken language often contains instances of non-standard forms, ranging from dialectal forms to neologisms, idiolectal forms, etc. For example:

(1) *it's a bit of fun, it livens up me day*

(2) *she told me to have them plums*

(3) *the dog is eat*

In these cases, the <seg> element will contain the non-standard form, and, optionally, provide indication of the target standard form.

[Note: It should be appreciated that assignment of a standard form is often a matter of choice by the annotator. Consider, for instance, the following examples:

a) *a man bought a horse and give it to her, now it's won the race*
b) *What I done, I taped it back like that*

In a), *give* might be corrected into *given* rather than *gave*, since frequently non standard English omits the auxiliary of the standard perfective construction. In b), *done* might be seen as equivalent either to standard *did* or to standard *have done*.

**Omissions**

Grammatically "necessary" words often happen to be omitted in naturally occurring dialogues. For example:

(4) *There's one thing I don't like and that's having my photo taken. And it will be hard when we have to photos.*

(5) *oh she was shouting at him at dinner time <shouting> Steven </shouting> oh god dinner time she was shouting him*

(6) *go in the sitting room until I shout you for tea*

(7) *The spelling mistakes only occurred when I was shouted*

In this case, it is necessary to identify as a <seg> element the orthographic words immediately preceding and following the omitted unit (see below, section 2.3.1.6).

**Incomplete utterances**

Incomplete (portions of) utterances frequently occur in spontaneous spoken dialogue, often because of interruptions. A main distinction can be drawn between a) interrupted sequences which are not followed by a completion, and b) interrupted sequences which are completed after the interruption, either by the same speaker or by another speaker.

An instance of the first type is the following:

(10) A: *smells good enough for*
A: *what is that?*
(11) *I wanted uh when is Margie coming?*

Instances of the second type are exemplified below:

- with self-completion:

(12) A: *so after the tower*
B: *yeah*
A: *I go straight ahead*

- with other-completion:

(13) A: *if Bill had known*
B: *he would have come*

Interrupted sequences of type a) are marked by a `<seg>` element only, further specified as broken (see below). Interrupted sequences of type b) are marked by means of two `<seg>` elements, one for the interrupted sequence and the other corresponding to the completion, much in the same vein of retrace-and-repair sequences (cf. next section).

**Retrace-and-repair sequences**

Instances of the phenomena falling into this wide category are:

Retracing without correction (simple repetitions):
(14) *I wanted I wanted to invite Margie*
(15) *It's it's it's it's like a um dog*

Retracing with correction:
(16) *I wanted uh I thought I wanted to invite Margie*
(17) *we're + at the same time we're real scared*

Retracing with reformulation:
(18) *All of my friends had uh we had decided to go home for lunch*
(19) *and that any bonus he # money he gets over that is a bonus*
(20) *That's why I said to get ma ba # get you back then*

All these phenomena are marked by two `<seg>` elements: the first one corresponding to the interrupted (repeated or reformulated) segment, the second one corresponding to the intended segment (or target form). Thus, in the example *I wanted I wanted to invite Margie* the first instance of the segment *I wanted* is to be viewed as a false start and the second instance as a target form. Similarly, in the example *I wanted uh when is Margie coming?* the first segment *I wanted* is

to be considered a false start and the second one, following the reformulation signal (*uh*), represents a target form. The reformulation signal itself can be tagged as an independent <seg> element; this strategy allows further specification of the structure of some types of dysfluency (see section 2.3.3).

#### 2.3.1.4 Assignment

Five different attributes are needed for the description of the element <seg>:

- **id**: a unique identifier
- **type**: whether the annotated segment is interrupted, or contains a non-standard form, or is followed by an omission, or else is a completion of a previously interrupted sequence
- **rep** (mnemonic for "replace"): a place for providing indication of the target form when the annotated segment contains a non-standard form
- **ins** (mnemonic for "insert"): a place for providing indication of the missing form when the annotated segment contains a gap.
- **href**: a sequence of <w> identifiers.

**Type**

Each identified <seg> element is further specified according to the particular type of dysfluency it represents. We have identified five possible values for this attribute: broken, sic, gap,scomp and ocomp.

broken qualifies a <seg> element as an interrupted portion of text, as is normally the case for the first sequence in interruptions, retracings, and false starts. The sic value serves to specify that a certain form contained in a <seg> element is non-standard. gap serves to mark a segment containing an omission. scomp and ocomp qualify a <seg> element as a completion of some previous interrupted sequence (which in turn will have been marked as broken): scomp indicate *self-completion* (i.e., the completion is uttered by the same speaker who was interrupted), ocomp indicates *other-completion* (i.e., the completion is uttered by the speaker who makes the interruption). For practical examples of use of the type attribute, see below, section 2.3.1.5.

| Values for `type` | Examples |
|---|---|
| `broken` | • [I wanted]`broken` I wanted to invite Margie<br>• [we're]`broken` at the same time we're real scared<br>• [All of my friends had]`broken` uh we had decided to go home for lunch<br>• [I wanted]`broken` uh when is Margie coming?<br>• A: [I'll do it if]`broken`<br>B: Yeah<br>A: you wish |
| `sic` | • it's a bit of fun, it livens up [me]`sic` day<br>• she told me to have [them]`sic` plums<br>• the dog is [eat]`sic` |
| `gap` | • oh she was shouting at him at dinner time <shouting> Steven </shouting> oh god dinner time she was [shouting him]`gap`<br>• go in the sitting room until I [shout you]`gap` for tea |
| `scomp` | • A: [I'll do it if]`broken`<br>B: Yeah<br>A: [you wish]`scomp` |
| `ocomp` | • A: [if Bill had known]`broken`<br>B: [he would have come]`ocomp` |

**Rep**

This attribute (mnemonic for "replace") allows the annotator to indicate the target or standard form of a non-standard usage. Note that the non standard form is to be assigned, in its turn, the value `sic` for the attribute `type`. In the example above, *it's a bit of fun, it livens up me day*, the segment corresponding to the word *me* would contain the string *my* as a value of the attribute `rep`.

It should be appreciated that assignment of a standard form is often a matter of choice by the annotator. Consider, for instance, the following examples:

(a) a man bought a horse and give it to her, now it's won the race
(b) What I done, I taped it back like that

In (a), *give* might be corrected into *given* rather than *gave*, since frequently non standard English omits the auxiliary of the standard perfective construction. In (b), *done* might be seen as equivalent either to standard *did* or to standard *have done*.

### Ins

The attribute `ins` (mnemonic for "insert") takes, as its value, a string signaling the part of speech of the omitted word (or sequence of words) when this cannot unambiguously be inferred from the context, as in the following example:

*vengo mangiare* ('I come eat')

where the preposition missing can be *a* ('to'), *per* ('for') or *da* ('from'). In this case, it is appropriate to signal the omission with no guess about the actually missing element. On the other hand, when it is possible to specify the actually omitted word, this is manually inserted as a child of the element `<seg>` (see below for examples).

### 2.3.1.5 Examples

**Non-standard forms**

(1) *it livens up me day*

orth.xml

```
<w id="w_001">it</w>
<w id="w_002">livens</w>
<w id="w_003">up</w>
<w id="w_004">me</w>
<w id="w_005">day</w>
```

edit.xml

```
<seg id="seg_001" type="sic" rep="my" href="orth.xml#id(w_004)"/>
```

**Non-standard forms**

(3) *the dog is eat*

orth.xml

```
<w id="w_001">the</w>
<w id="w_002">dog</w>
<w id="w_003">is</w>
<w id="w_004">eat</w>
```

edit.xml

```
<seg id="seg_001" type="sic" rep="eating" href="orth.xml#id(w_004)"/>
```

**Omissions**

(8) *go in the sitting room until I shout you for tea*

orth.xml

```
<w id="w_001">go</w>
<w id="w_002">in</w>
<w id="w 003">the</w>
```

```
<w id="w_004">sitting</w>
<w id="w_005">room</w>
<w id="w_006">until</w>
<w id="w_007">I</w>
<w id="w_008">shout</w>
<w id="w_009">you</w>
<w id="w_010">for</w>
<w id="w_011">tea</w>
```

edit.xml

```
<seg id="seg_001" type="gap" href="orth.xml#id(w_008)..id(w_009)">at</seg>
```

**Omissions**

*vengo mangiare*

orth.xml

```
<w id="w_001">vengo</w>
<w id="w_002">mangiare</w>
```

edit.xml

```
<seg id="seg_001" type="gap" ins="P" "orth.xml#id(w_001)..id(w_002)"/>
```

**Incomplete Utterances**

(10) *A: Smells good enough for*
*A: What is that?*

orth.xml

```
<u id="u_001" who="A">
  <w id="w_001">smells</w>
  <w id="w_002">good</w>
  <w id="w_003">enough</w>
  <w id="w_004">for</w>
</u>
<u id="u_002" who="A">
  <w id="w_005">what</w>
  <w id="w_006">is</w>
  <w id="w_007">that</w>
  <w id="w_008">?</w>
</u>
```

edit.xml

```
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_004)"/>
```

**Incomplete Utterances**

(11) *I wanted uh when is Margie coming?*

orth.xml

```
<w id="w_001">I</w>
<w id="w_002">wanted</w>
<w id="w_003">uh</w>
<w id="w_004">when</w>
<w id="w_005">is</w>
<w id="w_006">Margie</w>
<w id="w_007">coming</w>
<w id="w_008">?</w>
```

edit.xml

```
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_002)"/>
```

**Incomplete Utterances**

(12) A: *so after the tower*
B: *yeah*
A: *I go straight ahead*

orth.xml

```
<u id="u_001" who="A">
  <w id="w_001">so</w>
  <w id="w_002">after</w>
  <w id="w_003">the</w>
  <w id="w_004">tower</w>
</u>
<u id="u_002" who="B">
  <w id="w_005">yeah</w>
</u>
<u id="u_003" who="A">
  <w id="w_006">I</w>
  <w id="w_007">go</w>
  <w id="w_008">straight</w>
  <w id="w_009">ahead</w>
</u>
```

edit.xml

```
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_004)"/>
<seg id="seg_002" type="scomp"  href="orth.xml#id(w_006)..id(w_009)"/>
```

**Incomplete Utterances**

(13)A: *if Bill had known*
B: *he would have come*

orth.xml

```
...
<u id="u_001" who="A">
  <w id="w_001">if</w>
  <w id="w_002">Bill</w>
  <w id="w_003">had</w>
  <w id="w_004">known</w>
</u>
<u id="u_002" who="B">
  <w id="w_005">he</w>
  <w id="w_006">would</w>
  <w id="w_007">have</w>
  <w id="w_008">come</w>
</u>
...
```

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_004)"/>
<seg id="seg_002" type="ocomp"  href="orth.xml#id(w_005)..id(w_008)"/>
...
```

**Retrace-and-repair sequences**

(14) *I wanted I wanted to invite Margie*

orth.xml

```
...
<w id="w_001">I</w>
<w id="w_002">wanted</w>
<w id="w_003">I</w>
<w id="w_004">wanted</w>
<w id="w_005">to</w>
<w id="w_006">invite</w>
<w id="w_007">Margie</w>
...
```

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_002)"/>
<seg id="seg_002"               href="orth.xml#id(w_003)..id(w_007)"/>
...
```

**Retrace-and-repair sequences**

(15) *It's it's it's it's like a um dog*

orth.xml

```
...
<w id="w_001">it's</w>
<w id="w_002">it's</w>
<w id="w_003">it's</w>
<w id="w_004">it's</w>
<w id="w_005">like</w>
<w id="w_006">a</w>
<w id="w_007">um</w>
<w id="w_008">dog</w>
...
```

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_003)"/>
<seg id="seg_002"               href="orth.xml#id(w_004)"/>
...
```

**Retrace-and-repair sequences**

(16) *I wanted uh I thought I wanted to invite Margie*

orth.xml

```
...
<w id="w_001">I</w>
<w id="w_002">wanted</w>
<w id="w_003">uh</w>
<w id="w_004">I</w>
<w id="w_005">thought</w>
<w id="w_006">I</w>
<w id="w_007">wanted</w>
<w id="w_008">to</w>
<w id="w_009">invite</w>
<w id="w_010">Margie</w>
...
```

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_002)"/>
<seg id="seg_002"               href="orth.xml#id(w_004)..id(w_010)"/>
...
```

| **Retrace-and-repair sequences** |
| --- |
| (19) *That's why I said to get ma ba # get you back then* |

orth.xml

```
...
<w id="w_001">that's</w>
<w id="w_002">why</w>
<w id="w_003">I</w>
<w id="w_004">said</w>
<w id="w_005">to</w>
<w id="w_006">get</w>
<w id="w_007">my</w>
<w id="w_008">ba</w>
<w id="w_009">get</w>
<w id="w_010">you</w>
<w id="w_011">back</w>
<w id="w_012">then</w>
...
```

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_006)..id(w_008)"/>
<seg id="seg_002"               href="orth.xml#id(w_009)..id(w_012)"/>
...
```

#### 2.3.1.6 Markup Table

| <seg> | |
| --- | --- |
| id | [ASCII] |
| href | <w> |
| type | broken, sic, gap, scomp, ocomp |
| rep | [ASCII] |
| ins | [ASCII] |

### 2.3.2 <dys>: Dysfluencies

#### 2.3.2.1 Description

Annotation of these elements represents a recommended but not obligatory extension to the core annotation. <dys> elements serve the purpose of specifying the type of relationship between two <seg> elements, when these are used to mark dysfluencies of type a) (see section 2.3.1.3 above), i.e. repetitions, false starts, retracings and so on.

Let us remember that, in the core scheme described above, a dysfluency of type a) is marked by simply identifying the two portions of text which make up the dysfluency. This, however, is not further specified as to its type, i.e. whether it is a repetition, or a false start, or a retracing, etc.

<dys> elements logically presuppose prior identification of <seg> elements at the same level of annotation. Reference to <seg> elements is enforced through XML inline href links.

#### 2.3.2.2 Data Source

<dys> elements presuppose identification of <seg> elements at the level of edited transcription.

#### 2.3.2.3 Segmentation/Selection

A <dys> element is introduced by the annotator when the need is felt for further specifying the type of relationship between edited segments in the body of the transcript. Thus, <dys> elements are mainly put to use for augmenting annotation of dysfluencies of type a), and logically presuppose prior annotation of edited material as <seg> elements.

#### 2.3.2.4 Assignment

Three different attributes are needed for the description of the element <dys>:

- **id**: a unique identifier
- **type**: a categorization of the dysfluency
- **href**: a sequence of <seg> identifiers.

**Type**

This attribute is intended as a means for specifying the particular type of dysfluency represented by the element <dys>. In this scheme, no pre-defined value has been specified, for two reasons: first, a complete typology of dysfluent material is beyond the scope of the present meta-scheme; second, it might well be the case that the annotator wishes to use his/her own typology.

As an illustration, we provide here a handful of concrete examples of how the attribute type can be used. Values are based on CHILDES specifications (cf. MATE Deliverable D.1.1.).

**<dys> type: scomp (= self-completion)**

A: *so after the tower*

B: *yeah*

A: *I go straight ahead*

**<dys> type: ocomp (=other-completion)**

A: *if Bill had known*

B: *he would have come*

**<dys> type: repetition (simple)**

*I wanted I wanted to invite Margie*

*It's it's it's it's like a um dog*

**<dys> type: retrins (= retracing with insertion)**

*I wanted uh I thought I wanted to invite Margie*

*we're + at the same time we're real scared*

**<dys> type: retref (= retracing with reformulation)**

*All of my friends had uh we had decided to go home for lunch*

*and that any bonus he # money he gets over that is a bonus*

### 2.3.2.5 Examples

Annotation of the examples above (see section 2.3.1.6) could be optionally extended as follows:

**Self- and other completion**

(12) A: *so after the tower*
B: *yeah*
A: *I go straight ahead*

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_004)"/>
<seg id="seg_002" type="scomp"  href="orth.xml#id(w_006)..id(w_009)"/>
<dys id="dys_001" type="scomp"  href="edit.xml#id(seg_001)..id(seg_002)"/>
...
```

**Self- and other completion**

(13) A: *if Bill had known*
B: *he would have come*

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_004)"/>
<seg id="seg_002" type="ocomp"  href="orth.xml#id(w_005)..id(w_008)"/>
<dys id="dys_001" type="ocomp"  href="edit.xml#id(seg_001)..id(seg_002)"/>
...
```

**Simple repetitions**

(14) *I wanted I wanted to invite Margie*

edit.xml

```
...
<seg id="seg_001" type="broken"     href="orth.xml#id(w_001)..id(w_002)"/>
<seg id="seg_002"                   href="orth.xml#id(w_003)..id(w_007)"/>
<dys id="dys_001" type="repetition" href="edit.xml#id(seg_001)..id(seg_002)"/>
...
```

**Simple repetitions**

(15) *It’s it’s it’s it’s like a um dog*

edit.xml

```
...
<seg id="seg_001" type="broken"     href="orth.xml#id(w_001)..id(w_003)"/>
<seg id="seg_002"                   href="orth.xml#id(w_004)"/>
<dys id="dys_001" type="repetition" href="edit.xml#id(seg_001)..id(seg_002)"/>
...
```

**Retracing with insertion**

(16) *I wanted uh I thought I wanted to invite Margie*

edit.xml

```
...
<seg id="seg_001" type="broken"  href="orth.xml#id(w_001)..id(w_002)"/>
<seg id="seg_002"                href="orth.xml#id(w_004)..id(w_010)"/>
<dys id="dys_001" type="retrins" href="edit.xml#id(seg_001)..id(seg_002)"/>
...
```

**Retracing with reformulation**

(19) *That’s why I said to get ma ba # get you back then*

edit.xml

```
...
<seg id="seg_001" type="broken"  href="orth.xml#id(w_006)..id(w_008)"/>
<seg id="seg_002"                href="orth.xml#id(w_009)..id(w_012)"/>
<dys id="dys_001" type="retrref" href="edit.xml#id(seg_001)..id(seg_002)"/>
...
```

#### 2.3.2.6 Markup Table

| <dys> | |
|---|---|
| id | [ASCII] |
| href | <seg> |
| type | [ASCII] |

### 2.3.3 <reparandum>, <repair>, <signal>

#### 2.3.3.1 Description

These elements represent an optional extension to the core annotation, and they should merely be seen as an illustration of the expressive potential of the present meta-scheme. They are defined as hierarchical daughters of a <dys> element, and intended to represent components of retrace-and-repair sequences.

To be more concrete, consider the following example:

*I wanted uh I thought I wanted to invite Margie*

According to most approaches to the analysis of dysfluencies, the whole retrace-and-repair sequence can be further analyzed into a *reparandum* (the word sequence to be corrected or retraced), a *repair signal* (usually filled pauses or exclamations) and a *repair* (or correction, or alteration), as illustrated below:

*[I wanted]***reparandum** *[uh]***repair signal** *[I thought I wanted]***repair** *to invite Margie*

The elements <reparandum>, <signal> and <repair> should thus be seen as a means for a more detailed analysis of the components of a dysfluency. It should be noted that they refer to and qualify <seg> elements. They serve the purpose of specifying which previously identified <seg> element is repaired, which element is signalling that a repairing sequence is about to be uttered, and which element corresponds to the repair in the strict sense. This is achieved through pointing to already introduced <seg> elements through XML inline linking, and not by annotating orthographic words as such. From this it follows that if, say, the element signalling an interruption and an oncoming repair is not identified as a <seg> element, it is simply not possible to further mark it as <signal>.

This style of annotation allows overlaying of obligatory, recommended and optional layers of editing, and strikes a balance between two partially contradictory needs of i) leaving these layers mutually independent, and ii) achieving a maximally economic layering of editing information.

#### 2.3.3.2 Data Source

<reparandum>, <signal> and <repair> elements presuppose identification of <seg> elements at the level of edited transcription.

#### 2.3.3.3 Segmentation/Selection

These elements are introduced by the annotator when the need is felt for further specifying the structure of a dysfluency. Thus, these elements are mainly put to use for augmenting annotation of dysfluencies of type a), and logically presuppose prior annotation of edited material as <seg> elements. For an illustration of their use, see the examples below (cf. also section 2.3.3.1):

*[we're]* **reparandum** *[at the same time we're]* **repair** *real scared*

*[All of my friends had]* **reparandum** *[uh]* **repair signal** *[we had]* **repair** *decided to go home for lunch*

*[and that any bonus he]* **reparandum** *# [money he]* **repair** *gets over that is a bonus*

#### 2.3.3.4 Assignment

Two attributes are needed for the description of the elements <reparandum>, <signal> and <repair>:

- **id**: a unique identifier
- **href**: a sequence of <seg> identifiers.

#### 2.3.3.6 Examples

Annotation of the examples above (see section 2.3.2.6) could be optionally extended as follows:

**Simple repetitions**

(14) *I wanted I wanted to invite Margie*

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_002)"/>
<seg id="seg_002"               href="orth.xml#id(w_003)..id(w_007)"/>
<dys id="dys_001" type="repetition" href="edit.xml#id(seg_001)..id(seg_002)">
  <reparandum id="repm_001" href="edit.xml#id(seg_001)"/>
  <repair     id="rep_001"  href="edit.xml#id(seg_002)"/>
</dys>
...
```

**Simple repetitions**

(15) *It's it's it's it's like a um dog*

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_003)"/>
<seg id="seg_002"               href="orth.xml#id(w_004)"/>

<dys id="dys_001" type="repetition" href="edit.xml#id(seg_001)..id(seg_002)"/>
   <reparandum id="repm_001" href="edit.xml#id(seg_001)"/>
   <repair     id="rep_001"  href="edit.xml#id(seg_002)"/>
</dys>
...
```

**Retracing with insertion**

(16) *I wanted uh I thought I wanted to invite Margie*

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_001)..id(w_002)"/>
<seg id="seg_002"               href="orth.xml#id(w_003)"/>
<seg id="seg_003"               href="orth.xml#id(w_004)..id(w_010)"/>

<dys id="dys_001" type="retrins" href="edit.xml#id(seg_001)..id(seg_003)">
   <reparandum id="repm_001" href="edit.xml#id(seg_001)"/>
   <signal     id="sig_001"  href="edit.xml#id(seg_002)"/>
   <repair     id="rep_001"  href="edit.xml#id(seg_003)"/>
</dys>
...
```

| **Retracing with reformulation** |
| --- |
| (19) *That’s why I said to get ma ba # get you back then* |

edit.xml

```
...
<seg id="seg_001" type="broken" href="orth.xml#id(w_006)..id(w_008)"/>
<seg id="seg_002"               href="orth.xml#id(w_009)..id(w_012)"/>

<dys id="dys_001" type="retrref" href="edit.xml#id(seg_001)..id(seg_002)">
  <reparandum id="repm_001" href="edit.xml#id(seg_001)"/>
  <signal     id="sig_001"  href="edit.xml#id(seg_002)"/>
  <repair     id="rep_001"  href="edit.xml#id(seg_003)"/>
</dys>
...
```

### 2.3.3.5 Markup Table

| <reparandum> | |
| --- | --- |
| id | [ASCII] |
| href | <seg> |

| <signal> | |
| --- | --- |
| id | [ASCII] |
| href | <seg> |

| <repair> | |
| --- | --- |
| id | [ASCII] |
| href | <seg> |

# 3 Morphological Annotation Coding Module

## 3.1 Coding Purpose

Annotation of spoken dialogues at the word-level involves the following aspects:

- identification of morphological words
- annotation of part-of-speech categories
- annotation of morpho-syntactic features
- annotation of interrupted words
- annotation of cliticised words
- annotation of compounds
- annotation of morphological derivatives

## 3.2 Selected schemes

The meta-scheme outlined in the following pages is indebted to the formal specifications developed in EAGLES (see http://www.ilc.pi.cnr.it/EAGLES96), Multext (http://www.lpl.univ-aix.fr/projects/multext/) and CELEX (Burnage, 1990). Other aspects of word level annotation are also covered in the section on Edited Transcription.

## 3.3 Markup Declaration

```
<mw>
     <stem>
     <suffix>
     <prefix>
<cpw>
     <cpw_h>
```

### 3.3.1 <mw>: Morphological Words

#### 3.3.1.1 Description

<mw> elements serve to the markup of morphological words. As will be made clear below (see section 3.3.1.3), a morphological word stands, in the unmarked case, in a one-to-one relationship with an orthographic word. Exceptions to this tendency are represented by the case of many morphological words forming part of the same orthographic word (e.g. in cliticized words) and by

the case of many orthographic words which are in fact part of one and the same morphological word (e.g. in multi-word compounds).

Basically, morphological words are described at this level through the attributes `type` and `subtype`, which are intended to reflect EAGLES recommendations for morphosyntactic annotation. EAGLES recommendations make a distinction among three levels of specification, respectively encompassing *obligatory*, *recommended* and *optional* attribute-value pairs. Obligatory information concerns word category.

Word categories can further be specified by means of appropriate morphosyntactic features (such as gender, number, case etc.), expressed as supplementary recommended tags. The combination of a category tag with its morphosyntactic feature specification yields complex tags of considerable length and granularity, so as to cover the range of phenomena most commonly attested in European Languages. An outstanding set of optional tags are foreseen to deal with highly language-specific attribute-value pairs.

We partially mimic this three-fold level of recommended specifications as follows: the attribute type is intended to contain major obligatory part-of-speech categories, while the attribute subtype is used to introduce some recommended morpho-syntactic values. While the attribute type is specified obligatorily, the attribute subtype is only optional.

#### 3.3.1.2 Data Source

Tagging at the morphological level presupposes the markup of orthographic words.

#### 3.3.1.3 Segmentation/Selection

As already observed above, in many cases, orthographic and morphological words are in a one-to-one correspondence. There are several exceptions to this tendency, however, as shown by the existence of compounds such as *credit card*, or cliticised words such as Italian *dimmelo* 'say it to me'. In the former case, two distinct orthographic words make up a unique compounded word; in the latter case, a unique orthographic word is made up out of three distinct morphological words, namely *di* 'say', *mi* 'to me' and *lo* 'it'.

From this it follows that annotation at the morphological level involves at least three different labelling mechanisms: i) marking up one word according to a set of morpholexical and morphosyntactic features; this represents the simplest case, where one morphological word corresponds to one orthographic word only; ii) grouping more than one orthographic word into one compounded constituent; this is the case of annotating a compound like *credit card*; iii) segmenting a morphologically complex orthographic word into its constituent elements, as in the case of cliticised words, such as Italian *diglielo, fammi, fammelo,* Spanish *daselo, diselo,* but also English *it's, lemme.* Note that, in case ii) above, a single morphological word will point to a sequence of two or more orthographic words in the resource file. In the case of cliticised words, on the other hand, two or more morphological words will point to the same orthographic form. In this latter case, a standardized form of the orthographic counterparts of the identified morphological words will be specified by manually inserting it as a CDATA child of the `<mw>` element being annotated, (see the examples below).

In simple cases, this standard form coincides with a substring of the annotated word form: for example the cliticised form daselo is to be segmented as da-se-lo, where the segments da, se and lo are also found as independent word forms in Spanish. In other cases, the standard form abstracts away from the orthographic segment, when the latter undergoes changes such as elision, epenthesis etc. For example, the form it's should be segmented into the standard forms it and is, and not - say - into it and 's.

These annotation activities are not necessarily mutually exclusive, as it is often the case that segmented constituents are, in their turn, to be specified for their morphosyntactic features etc. In this document, we provide a unified mark-up framework for carrying out the required annotation practices in an integrated way.

#### 3.3.1.4 Assignment

Four attributes are obligatorily needed for the description of morphological words:

- **id**: a unique identifier
- **href**: one or more `<w>` identifiers
- **type**: the part-of-speech category, according to EAGLES specifications
- **lemma**: specifies the lemma

Optional attributes are the following:

- **subtype**: additional morpho-syntactic features
- **broken:** whether the word is complete or interrupted

**Type**

Each identified morphological word must be labelled as to its part-of-speech category. POS categories, to be expressed as values of the type attribute, are taken from the set of morphosyntactic major categories detailed in EAGLES specifications for morpho-syntactic annotation. The table below summarizes the entire range of values.

| VALUE | POS |
|---|---|
| N | noun |
| V | verb |
| AJ | adjective |
| PD | pronoun/determiner |
| AT | article |
| AV | adverb |

| | |
|---|---|
| AP | adposition |
| C | conjunction |
| NU | numeral |
| I | interjection |
| U | unique/unassigned |
| R | residual |
| F | filler |
| DM | discourse marker |
| PU | punctuation |

Most of the above tags are self-explanatory; a few words about the following tags are needed (quoting from the relevant EAGLES document):

> The unique value is applied to categories with a unique or very small membership, such as negative particles, which are "unassigned" to any of the non standard part-of-speech categories.
>
> The residual values is assigned to words which lie outside the traditionally accepted range of grammatical classes, although they occur quite commonly (for example: foreign words).

The above tag set is a slight modification of the original EAGLES tag set in that it introduces two new values: F (for fillers) and DM (for discourse markers). They are intended to annotate phenomena that, although not specific of spoken language, are far more common in spoken dialogue than they are in writing. Fillers are defined here as all instances of non-lexical vocalizations typically uttered in filled pauses (e.g., *um*, *uhm*, *mh-mm*, etc.).
Discourse markers or discourse items are adverbs, conjuctions and small clauses such as *well, you know*, *I mean*, and so on, used as interactional markers (turn-taking and giving, signals of correction, understanding, prompting, and connection to previous utterances).

### Lemma

This attribute is intended to provide a place for specifying the exponent form of the lemma of the annotated word form, if a separate lexicon in XML format is not available. For example, the lemma of forms such as *is* and *are* is the bare infinitive *be* by lexical convention. This attribute should not be used if such a lexicon is available; in this case, an alternative annotation is applied instead (see section 5.5.1).

### Subtype

This attribute specifies additional recommended morphosyntactic features, according to EAGLES specifications for morpho-syntactic annotation. A value of the attribute subtype is in fact a complex sequence of atomic symbols to be interpreted according to a specific grid of positional

slots. In EAGLES, each major morphosyntactic category is associated with a specific grid of positions. For example, the grid for verb-specific morphosyntactic features is given in the following table:

| | | | | | |
|---|---|---|---|---|---|
| (i) | Person: | 1. First | 2. Second | 3. Third | |
| (ii) | Gender: | 1. Masculine | 2. Feminine | 3. Neuter | |
| (iii) | Number: | 1. Singular | 2. Plural | | |
| (iv) | Finiteness: | 1. Finite | 2. Non-finite | | |
| (v) | Verbform/ Mood: | 1. Indicative | 2. Subjective | 3. Imperative | 4. Conditional |
| | | 5. Infinite | 6. Participle | 7. Gerund | 8. Supine |
| (vi) | Tense: | 1. Present | 2. Imperfect | 3. Future | 4. Past |
| (vii) | Voice: | 1. Active | 2. Passive | | |
| (viii) | Status: | 1. Main | 2. Auxiliary | | |

An Italian verb form such as *andò* '(he) went', for example, which conveys the values '3rd person, singular, finite, indicative, past tense, active, main verb, non-phrasal, non-reflexive, verb' would be represented, according to the table above, as the complex value '**V3011141101200'**.

Wherever an attribute is inapplicable to a given word in a given tagset, the value **0** fills that attribute position in the string of digits. When the **0**s occur in final position, without any non-zero digits following, they can be dropped.

The complete set of EAGLES positional grids for any part of speech is provided as an appendix to this document (cf. the Appendix).

**Broken**: the word is interrupted (Y) or not (N, default value).

3.3.1.5 Examples

(1) *We take the oranges to Elmira uh I mean to Corning*

orth.xml

```
...
<w id="w_001">we</w>
<w id="w_002">take</w>
<w id="w_003">the</w>
<w id="w_004">oranges</w>
<w id="w_005">to</w>
<w id="w_006">Elmira</w>
<w id="w_007">uh</w>
<w id="w_008">I</w>
<w id="w_009">mean</w>
<w id="w_010">to</w>
<w id="w_011">Corning</w>
...
```

mword.xml

```
...
<mw id="mw_001" type="PD" subtype="PD1020115" lemma="we"      href="orth.xml#id(w_001)"/>
<mw id="mw_002" type="V"  subtype="V00011111" lemma="take"    href="orth.xml#id(w_002)"/>
<mw id="mw_003" type="AT" subtype="AT1000"    lemma="the"     href="orth.xml#id(w_003)"/>
<mw id="mw_004" type="N"  subtype="N102000"   lemma="orange"  href="orth.xml#id(w_004)"/>
<mw id="mw_005" type="AP" subtype="AP1"       lemma="to"      href="orth.xml#id(w_005)"/>
<mw id="mw_006" type="N"  subtype="N201000"   lemma="Elmira"  href="orth.xml#id(w_006)"/>
<mw id="mw_007" type="I"  subtype="I"                         href="orth.xml#id(w_007)"/>
<mw id="mw_008" type="PD" subtype="PD1010115" lemma="I"       href="orth.xml#id(w_008)"/>
<mw id="mw_009" type="V"  subtype="V00011111" lemma="mean"    href="orth.xml#id(w_009)"/>
<mw id="mw_010" type="AP" subtype="AP1"       lemma="to"      href="orth.xml#id(w_010)"/>
<mw id="mw_011" type="N"  subtype="N201000"   lemma="Corning" href="orth.xml#id(w_011)"/>
<mw id="mw_001" type="PD" subtype="PD1020115" lemma="we"      href="orth.xml#id(w_001)"/>
<mw id="mw_002" type="V"  subtype="V00011111" lemma="take"    href="orth.xml#id(w_002)"/>
<mw id="mw_003" type="AT" subtype="AT1000"    lemma="the"     href="orth.xml#id(w_003)"/>
<mw id="mw_004" type="N"  subtype="N102000"   lemma="orange"  href="orth.xml#id(w_004)"/>
<mw id="mw_005" type="AP" subtype="AP1"       lemma="to"      href="orth.xml#id(w_005)"/>
<mw id="mw_006" type="N"  subtype="N201000"   lemma="Elmira"  href="orth.xml#id(w_006)"/>
<mw id="mw_007" type="I"  subtype="I"                         href="orth.xml#id(w_007)"/>
<mw id="mw_008" type="PD" subtype="PD1010115" lemma="I"       href="orth.xml#id(w_008)"/>
<mw id="mw_009" type="V"  subtype="V00011111" lemma="mean"    href="orth.xml#id(w_009)"/>
<mw id="mw_010" type="AP" subtype="AP1"       lemma="to"      href="orth.xml#id(w_010)"/>
<mw id="mw_011" type="N"  subtype="N201000"   lemma="Corning" href="orth.xml#id(w_011)"/>
...
```

(2) *diglielo*

orth.xml

```
...
<w id="w_001"> diglielo </w>
...
```

mword.xml

```
...
<mw id="mw_001" type="V"  subtype="V20113101" lemma="dire" href="orth.xml#id(w_001)">di</mw>
<mw id="mw_002" type="PD" subtype="PD3110315" lemma="egli" href="orth.xml#id(w_001)">gli</mw>
<mw id="mw_003" type="PD" subtype="PD3110415" lemma="esso" href="orth.xml#id(w_001)">lo</mw>
...
```

(3) *fammi*

orth.xml

```
...
<w id="w_001">fammi</w>
...
```

mword.xml

```
...
<mw id="mw_001" type="V"  subtype="V20113101" lemma="fare" href="orth.xml#id(w_001)">fa</mw>
<mw id="mw_002" type="PD" subtype="PD1010315" lemma="io"   href="orth.xml#id(w_001)">mi</mw>
...
```

(4) *glielo dico*

orth.xml

```
...
<w id="w_001">glielo</w>
```

```
<w id="w_002">dico</w>
...
```

mword.xml

```
...
<mw id="mw_001" type="PD" subtype="PD3110315" lemma="egli" href="orth.xml#id(w_001)">gli</mw>
<mw id="mw_002" type="PD" subtype="PD3110415" lemma="esso" href="orth.xml#id(w_001)">lo</mw>
<mw id="mw_003" type="V"  subtype="V10111101" lemma="dire" href="orth.xml#id(w_002)"/>
...
```

(5) *red skin*

orth.xml

```
...
<w id="w_001">red</w>
<w id="w_002">skin</w>
...
```

mword.xml

```
...
<mw id="mw_001" type="N" subtype="N101000" href="orth.xml#id(w_001)..id(w_002)"/>
...
```

(6) *I know it's late*

orth.xml

```
...
<w id="w_001">I</w>
<w id="w_002">know</w>
<w id="w_003">it's</w>
<w id="w_004">late</w>
...
```

mword.xml

```
...
<mw id="mw_001" type="PD" subtype="PD1010115"  lemma="I"    href="orth.xml#id(w_001)"/>
<mw id="mw_002" type="V"  subtype="V00011111"  lemma="know" href="orth.xml#id(w_002)"/>
<mw id="mw_003" type="PD" subtype="PD33101151" lemma="it"   href="orth.xml#id(w_003)">it</mw>
<mw id="mw_004" type="V"  subtype="V30111111"  lemma="be"   href="orth.xml#id(w_003)">is</mw>
<mw id="mw_005" type="AV" subtype="AV1120"     lemma="late" href="orth.xml#id(w_004)"/>
...
```

### 3.3.1.6 Markup Table

| <mw> | |
|---|---|
| id | [ASCII] |
| href | <w> |
| type | N, V, AJ, PD, AT, AV, AP, C, NU, I, U, R, F, DM, PU |
| subtype | (cfr. EAGLES tables) |
| lemma | [ASCII] |
| broken | Y, N |

### 3.3.2 <cpw>: Compound Words

#### 3.3.2.1 Description

Compounds are annotated using <cpw> elements. <cpw> elements are multi-word units whose constituents are linked to morphological words through in-line reference. Thus, unlike any other element identified at this level of annotation, <cpw> elements are linked only indirectly to orthographic words (via morphological words). This is required by the specific linguistic nature of compounds, whose syntactic behaviour is in many respects completely independent of (or opaque to) their internal complex structure, which is nonetheless analysed in terms of (possibly recursive) levels of embedding. This hybrid status forces the annotator to annotate morphological constituents first, to then tag them as forming part of the same morphological construct.

Compounds are represented orthographically in a variety of different ways, ranging from a one word representation, as is commonly the case in German, and more rarely in English and Italian compounds (*cupboard*, *blackbird*, *cassaforte* 'safe' etc.), to a dashed multiword unit (as in *common-or-garden*), to a sequence of independent orthographic words (as in *credit card* or Italian "syntactic" compounds such as *ferro da stiro* 'iron'). As orthographic representations are crucially a matter of convention and do not line up with linguistically grounded distinctions, it is recommended that annotation of compounds abstract away from considerations concerning orthography, and be motivated only on linguistic grounds. In the following, we will provide such a skeletal linguistically-based typology and suggest ways of representing it through XML. It is important to notice, however, two things. First, the encoding conventions suggested here for compounds can be put to use to annotate other linguistic material which does not fall traditionally into the category of compounds. This material includes frozen expressions such as *ad hoc, a priori, matter of fact* etc., or complex dialogue markers such as *you know*. Secondly, one word compounds should not be annotated as chains of segmented constituents as suggested for morphological derivatives. If one does so, it would miss out a lot of information normally associated with the identification of compounds.

#### 3.3.2.2 Data Source

Tagging of <cpw> presupposes the markup of morphological words.

#### 3.3.2.3 Segmentation/Selection

Although compounding represents a critical area for both theoretical and computational morphology, annotation of compounds (as opposed to their identification or their interpretation) can be a relatively trivial issue if limited to signalling membership of a sequence of word forms (such as *copy* and *editor* in *copy editor*, or *ferro + da + stiro* in *ferro da stiro*) to a morphosyntactically unique word. Concrete identification of these constituents may vary depending on the orthographic rendering of a compound. In cases such as English *cupboard* or *pineapple*, identification of the constituents of a compound requires the process of singling out word-internal constituents, instead of grouping independent orthographic words. In still further cases, segmentation is already indirectly signalled in the orthography by means of dashes: as in *common-or-garden*. Since orthography is a rather poor indicator of the morphological nature of a compound to be annotated, we suggest that the representation of compounds be grounded on linguistic motivations only, as sketchily suggested in the following.

A useful distinction to be made is that between endocentric and exocentric compounds, the former showing a semantic head (e.g. *mother milk* is a kind of *milk*), which is not present in the latter (e.g. *red skin* or *redskin* is not a kind of *skin*). Due to their non compositionality at the level of meaning, it is recommendable that exocentric compounds be treated as whole morphological words. Note that also other types of multiword units which are not commonly categorised as compounds, such as, for example, discourse markers like *you know*, can receive a representation as whole morphological words.

There is wider annotation leeway in the case of endocentric compounds, where it can be argued that the constituent *mother* in *mother milk* is a sort of modifier of the head *milk*, both syntactically and semantically. Since we mark modifiers as independent units at the functional level, it may be convenient to keep the two constituents of the compound *mother milk* separate at the level of morphological analysis. Still we are interested in annotating the fact that *mother milk* is a compound. The mark-up scheme suggested in the following section makes provision for both kinds of solution. A `<head>` element (see below) is contained by each `<cpw>` element and serves to specify the semantic head of the compound.

As a rule, we recommend that endocentric compounds be annotated via `<cpw>` elements, and exocentric compounds be treated as a single morphological unit.

Clearly, provided that consistency is assured, nothing opposes to annotating all compounds as one morphological word, or conversely to using a `<cpw>` element for the annotation of any compound, be it endo- or exo-centric.

#### 3.3.2.4 Assignment

Five different attributes are needed for the description of compound words:

- **id**: a unique identifier
- **href**: one or more `<mw>` identifiers
- **type**: the part-of-speech category, according to EAGLES specifications
- **subtype**: additional morpho-syntactic features
- **broken**: specifies whether the word is complete (default value) or interrupted

For criteria of assignment of the attributes `type` and `subtype`, see above, section 3.3.1.4.

#### 3.3.2.5 Examples

(1) *mother milk*

orth.xml

```
...
<w id="w_001">mother</w>
<w id="w_002">milk</w>
...
```

mword.xml

```
...
<mw id="mw_001" type="N" subtype="N10101" lemma="mother"
```

```
                                        href="orth.xml#id(w_001)"/>
<mw id="mw_002" type="N" subtype="N10102" lemma="milk"
                                        href="orth.xml#id(w_002)"/>

<cpw id="cpw_001" type="N" href="mword.xml#id(mw_001)..id(mw_002)"/>
...
```

(2) *credit card*

orth.xml

```
...
<w id="w_001">credit</w>
<w id="w_002">card</w>
...
```

mword.xml

```
...
<mw id="mw_001" type="N" subtype="N10101" lemma="credit" href="orth.xml#id(w_001)"/>
<mw id="mw_002" type="N" subtype="N10101" lemma="card"   href="orth.xml#id(w_002)"/>

<cpw id="cpw_001" type="N" href="mword.xml#id(mw_001)..id(mw_002)"/>
...
```

#### 3.3.2.6 Markup Table

| <cpw> | |
|---|---|
| id | [ASCII] |
| href | <mw> |
| type | N, V, AJ, PD, AT, AV, AP, C, NU, I, U, R, F, DM |
| subtype | (cfr. EAGLES specifications) |
| broken | Y, N |

### 3.3.3 <cpw_ h>: Heads in Compound Words

#### 3.3.3.1 Description

A <cpw_h> element is used to mark the semantic head in a compound. As a rule of thumb, headed compounds (or endocentric compounds) should always be in an IS_A relationship to their heads: e.g., a *school bus* is_a bus, a *credit card* is a *card* etc.

#### 3.3.3.2 Data Source

For the tagging of heads, the markup of morphological words is necessary.

#### 3.3.3.3 Segmentation/Selection

It is common knowledge that the head of a compound always takes an obligatory position in the sequence of word constituents. This position is subject to language-dependent parameterisation: in languages such as German or English, the head of a compound is the rightmost word element in the chain; in French or Italian the head normally takes the leftmost position, although there can be exceptions to this general tendency, due to the analogical pressure of left-headed compounds (as in Italian *scuola bus* 'school bus' instead of the expected *bus scuola*).

#### 3.3.3.4 Assignment

Two attributes are needed for the description of head elements:

- **id**: a unique identifier
- **href**: one <mw> identifier

#### 3.3.3.5 Examples

(1) *school bus*

orth.xml

```
...
<w id="w_001">school</w>
<w id="w_002">bus</w>
...
```

mword.xml

```
...
<mw id="mw_001" type="N" subtype="N10101" href="orth.xml#id(w_001)"/>
<mw id="mw_002" type="N" subtype="N10101" href="orth.xml#id(w_002)"/>

<cpw     id="cpw_001"  type="N" href="mword.xml#id(mw_001)..id(mw_002)">
  <cpw_h id="cpwh_001"          href="mword.xml#id(mw_002)"/>
</cpw>
...
```

(2) *credit card*

orth.xml

```
...
<w id="w_001">credit</w>
<w id="w_002">card</w>
...
```

mword.xml

```
...
<mw id="mw_001" type="N" subtype="N10101" href="orth.xml#id(w_001)"/>
<mw id="mw_002" type="N" subtype="N10101" href="orth.xml#id(w_002)"/>
<cpw     id="cpw_001"  type="N" href="mword.xml#id(mw_001)..id(mw_002)">
  <cpw_h id="cpwh_001"          href="mword.xml#id(mw_002)"/>
</cpw>
...
```

#### 3.3.3.6 Markup Table

| <cpw_h> |
|---|

| | |
|---|---|
| id | [ASCII] |
| href | <mw> |

### 3.3.4 <stem>, <prefix>, <suffix>: Derivational Morphology

#### 3.3.4.1 Description

This set of elements is used to annotate derivational morphology. Unlike inflected forms, derivatives such as Italian *derivazion-ale*, or English *derivation-al, frank-ly, friend-ly*, lend themselves more naturally to being marked up as a chain of segmented constituents. "Morpheme segmentation", either immediate (e.g. signalling the most external affix only, as in "derivation-al"), or complete (as in "deriv-ation-al") or hierarchical (as in "(((deriv) ation) al)") is provided, for example, in the CELEX electronic lexica (Burnage, 1990). Yet, this type of representation is, in general, not able to account for non concatenative phenomena such as stem allomorphy (admittedly far less frequent in derivational morphology than in inflectional morphology). For lack of better consensual encoding practices, immediate flat morpheme segmentation could be proposed as a reasonable minimal annotation strategy for encoding derivational morphemes. Note that segmentation is represented here rather indirectly, that is, not through interspersion of dashes in the orthographic rendering of a derivative, but through indication of the standard form of the internal constituents of the derivative. The standard form is expressed, as in the case of cliticised words, as a value of the attribute `seg`, often not the lemma, but the corresponding form found as an independent orthographic unit. This definition, however, does not apply to the case of bound morphemes such as derivational suffixes/prefixes. In this case, we suggest, as a rule of thumb, to assign to `seg` a base form corresponding to the orthography of the suffix/prefix as it shows up in non fused environments. In difficult cases, one can resort to more than one base form.

#### 3.3.4.2 Data Source

Annotation of derivational morphology presupposes the markup of orthographic words.

#### 3.3.4.3 Segmentation/Selection

Flat immediate segmentation requires identification of the most external suffix/prefix in the derivative: e.g. *industri-al*, *considerab-ly* etc. When a derivative is both prefixed and suffixed, criteria of selectional restrictions on the way morphemes are concatenated are normally invoked to establish whether the newly added morpheme is a prefix or a suffix: e.g. in a word such as *recognition*, the suffix *-ion* is uncontroversially the last morpheme, since the prefix *re-* can only be attached to verb bases/roots (*re-cognise*), but not to nouns (**re-cognition*). In other more ambiguous cases, the annotator must be guided by lexico-semantic criteria: e.g., the morpheme *anti-* can be prefixed to both nouns (*anti-missile, anti-matter*) and adjectives (*anti-social*). In a case such as *anti-semitic*, it is the existence of the noun *anti-semite* and its strong semantic relation with *anti-semitic* which favours consideration of the suffix *-ic* as the last morpheme, attached to the base *anti-semite*.

It should be noted that the style of XML representation illustrated in the following allows the annotator to get around a number of paradoxes of orthographic segmentation. In fact, word-internal constituents are represented not as strings separated by dashes, but rather as standard

forms which are represented as CDATA children of the <stem>, <suffix> and <prefix> elements.
In cases of highly fused derivatives, such as, for example, *recognition*, we suggest to assign the standard forms *recognise* and *ion* to the two identified constituents of the immediate segmentation. This makes provision for some kind of abstract representation, thus avoiding the problem of defining an appropriate segmentation of the whole orthographic form *recognition*, which is fairly controversial.

#### 3.3.4.4 Assignment

Two attributes are needed for the description of stems, suffixes and prefixes:

- **id**: a unique identifier
- **href**: one <mw> identifier

The element <stem> can be further specified for an attribute "type", by means of which it is possible to state whether the stem is verbal or nominal.

#### 3.3.4.5 Examples

(1) *derivational morphology*

orth.xml

```
...
<w id="w_001">derivational</w>
<w id="w_002">morphology</w>
...
```

mword.xml

```
...
<mw       id="mw_001" type="AJ" href="orth.xml#id(w_001)">
  <stem   id="st_001" type="N"  href="orth.xml#id(w_001)">derivation</stem>
  <suffix id="su_001"           href="orth.xml#id(w_001)">al</suffix>
</mw>
<mw       id="mw_002" type="N"  href="orth.xml#id(w_002)"/>
...
```

(2) *frankly*

orth.xml

```
...
<w id="w_001">frankly</w>
...
```

mword.xml

```
...
<mw       id="mw_001" type="AV" href="orth.xml#id(w_001)">
  <stem   id="st_001" type="AJ" href="orth.xml#id(w_001)">frank</stem>
  <suffix id="su_001"           href="orth.xml#id(w_001)">ly</suffix>
</mw>
...
```

#### 3.3.4.5 Markup Table

| <stem> | |
|---|---|
| id | [ASCII] |
| href | <w> |
| type | N, V |
| **<suffix>** | |
| id | [ASCII] |
| href | <w> |
| **<prefix>** | |
| id | [ASCII] |
| href | <w> |

Annotation at the chunking level

Hereafter we provide an example of annotation at the chunking level.

chunk.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE chunk_file SYSTEM "chunk.dtd">
<chunk_file>
<ch id="ch_001" type="FV"   href="mword#id(mw_001)">
  <potgov id="p_001"        href="mword.xml#id(mw_001)"/>
</ch>
<ch id="ch_002" type="N"    href="mword.xml#id(mw_002)..id(mw_003)">
  <potgov id="p_002"        href="mword.xml#id(mw_003)"/>
</ch>
<ch id="ch_003" type="I"    href="mword.xml#id(mw_004)..id(mw_005)">
  <potgov id="p_003"        href="mword.xml#id(mw_005)"/>
</ch>
<ch id="ch_004" type="N"    href="mword.xml#id(mw_006)..id(mw_007)">
  <potgov id="p_004"        href="mword.xml#id(mw_007)"/>
</ch>
<ch id="ch_005" type="P"    href="mword.xml#id(mw_008)..id(mw_009)">
  <potgov id="p_005"        href="mword.xml#id(mw_009)"/>
</ch>
<ch id="ch_006" type="P"    href="mword.xml#id(mw_010)..id(mw_011)">
  <potgov id="p_006"        href="mword.xml#id(mw_011)"/>
</ch>
<ch id="ch_007" type="ADV"  href="mword.xml#id(mw_012)">
  <potgov id="p_007"        href="mword.xml#id(mw_012)"/>
</ch>
<ch id="ch_008" type="ADV"  href="mword.xml#id(mw_013)">
  <potgov id="p_008"        href="mword.xml#id(mw_013)"/>
</ch>
<ch id="ch_009" type="FV"   href="mword.xml#id(mw_014)">
  <potgov id="p_009         href="mword.xml#id(mw_014)"/>
</ch>
<ch id="ch_010" type="N"    href="mword.xml#id(mw_015)..id(mw_016)">
  <potgov id="p_010"        href="mword.xml#id(mw_016)"/>
</ch>
<ch id="ch_011" type="PART" href="mword.xml#id(mw_017)">
  <potgov id="p_011"        href="mword.xml#id(mw_017)"/>
</ch>
<ch id="ch_012" type="N"    href="mword.xml#id(mw_018)">
  <potgov id="p_012"        href="mword.xml#id(mw_018)"/>
</ch>
<ch id="ch_013" type="FV"   href="mword.xml#id(mw_019)..id(mw_021)">
  <potgov id="p_013"        href="mword.xml#id(mw_021)"/>
  <aux id="aux_001"         href="mword.xml#id(mw_019)"/>
```

```
</ch>
<ch id="ch_014" type="N"    href="mword.xml#id(mw_022)..id(mw_023)">
  <potgov id="p_014"        href="mword.xml#id(mw_023)"/>
</ch>
<ch id="ch_015" type="P"    href="mword.xml#id(mw_024)..id(mw_026)">
  <potgov id="p_015"        href="mword.xml#id(mw_026)"/>
  <intro id="i_0o1"         href="mword.xml#id(mw_024)"/>
</ch>
<ch id="ch_016" type="PART" href="mword.xml#id(mw_027)">
  <potgov id="p_016"        href="mword.xml#id(mw_027)"/>
</ch>
<ch id="ch_017" type="ADV"  href="mword.xml#id(mw_028)">
  <potgov id="p_017"        href="mword.xml#id(mw_028)"/>
</ch>
<ch id="ch_018" type="FV"   href="mword.xml#id(mw_029)">
  <potgov id="p_018"        href="mword.xml#id(mw_029)"/>
</ch>
<ch id="ch_019" type="N"    href="mword.xml#id(mw_030)..id(mw_031)">
  <potgov id="p_019"        href="mword.xml#id(mw_031)"/>
</ch>
<ch id="ch_020" type="ADV"  href="mword.xml#id(mw_032)">
  <potgov id="p_020"        href="mword.xml#id(mw_032)"/>
</ch>
<ch id="ch_021" type="FV"   href="mword.xml#id(mw_033)">
  <potgov id="p_021"        href="mword.xml#id(mw_033)"/>
</ch>
<ch id="ch_022" type="N"    href="mword.xml#id(mw_034)..id(mw_035)">
  <potgov id="p_022"        href="mword.xml#id(mw_035)"/>
</ch>
<ch id="ch_023" type="P"    href="mword.xml#id(mw_036)..id(mw_037)">
  <potgov id="p_023"        href="mword.xml#id(mw_037)"/>
<intro id="i_002"           href="mword.xml#id(mw_036)"/>
</ch>
<ch id="ch_024" type="P"    href="mword.xml#id(mw_038)..id(mw_039)">
  <potgov id="p_024"        href="mword.xml#id(mw_039)"/>
<intro id="i_003"           href="mword.xml#id(mw_038)"/>
</ch>
<ch id="ch_025" type="ADV"  href="mword.xml#id(mw_040)">
  <potgov id="p_025"        href="mword.xml#id(mw_040)"/>
</ch>
<ch id="ch_026" type="FV"   href="mword.xml#id(mw_041)">
  <potgov id="p_026"        href="mword.xml#id(mw_041)"/>
</ch>
<ch id="ch_027" type="N"    href="mword.xml#id(mw_042)..id(mw_044)">
  <potgov id="p_027"        href="mword.xml#id(mw_044)"/>
</ch>
<ch id="ch_028" type="ADV"  href="mword.xml#id(mw_045)">
  <potgov id="p_0248"       href="mword.xml#id(mw_045)"/>
</ch>
<ch id="ch_029" type="FV"   href="mword.xml#id(mw_046)">
  <potgov id="p_029"        href="mword.xml#id(mw_046)"/>
</ch>
<ch id="ch_030" type="N"    href="mword.xml#id(mw_047)..id(mw_048)">
  <potgov id="p_030"        href="mword.xml#id(mw_048)"/>
</ch>
<ch id="ch_031" type="P"    href="mword.xml#id(mw_049)..id(mw_050)">
  <potgov id="p_031"        href="mword.xml#id(mw_050)"/>
  <intro id="i_004"         href="mword.xml#id(mw_049)"/>
</ch>
<ch id="ch_032" type="P"    href="mword.xml#id(mw_051)..id(mw_052)">
  <potgov id="p_032"        href="mword.xml#id(mw_052)"/>
  <intro id="i_005"         href="mword.xml#id(mw_051)"/>
</ch>
<ch id="ch_033" type="N"    href="mword.xml#id(mw_054)">
  <potgov id="p_033"        href="mword.xml#id(mw_054)"/>
</ch>
<ch id="ch_034" type="P"    href="mword.xml#id(mw_055)..id(mw_056)">
  <potgov id="p_034"        href="mword.xml#id(mw_056)"/>
  <intro id="i_006"         href="mword.xml#id(mw_055)"/>
</ch>
</chunk_file>
```

# 4 Chunk-level Annotation Coding Module

## 4.1 Coding Purpose

This section is concerned with syntactic annotation at the chunking level. Chunks are textual units of adjacent word tokens which can be linked mutually through unambiguously identified dependency chains with no recourse to idiosyncratic lexical information.

In marking chunks, we are mainly interested in their category and start and end points. It should be noted that chunks do not necessarily cover the entire sentence, as there may be material that does not belong to any chunk. For example, prepositions, coordinators, subordinators, and adverbs are, in some cases and according to some instantiations of chunking, not part of any chunk.

## 4.2 Selected schemes

There are several approaches to chunking, which comply with somewhat varying notions of what a chunk is (cf. Abney, 1996; Federici, Montemagni and Pirrelli, 1996, 1998). The notation presented here is based on SPARKLE specifications (Carroll et al., 1997), which were developed so as to represent an edited intersection of different existing schemes.

## 4.3 Markup Declaration

```
<ch>
          <potgov>
          <aux>
          <cop>
          <intro>
          <modal>
          <caus>
```

### 4.3.1 <ch>: Chunks

#### 4.3.1.1 Description

Chunks are defined strictly synctactically. Following Abney (1996:1), a chunk is "the non-recursive core of an intra-clausal constituent, extending from the beginning of a constituent to its head (or *potential governor*, see below), but not including post-head dependents".

Each chunk includes a sequence of adjacent word tokens which are mutually related through dependency links. For a more detailed discussion, cf. MATE Deliverable D.1.1. Examples and criteria for chunking are given in the following sections.

#### 4.3.1.2 Data Source

Tagging at the chunk level presupposes the markup of morphological words.

#### 4.3.1.3 Segmentation/Selection

Given the sentence "The hungry man could always eat the meals offered by the pious woman", the chunking will be as follows:

[*The hungry man*] [*could always eat*] [*the meals*] [*offered*] [*by the pious woman*].

The sentence is segmented into five chunks. As noted before, each chunk includes a sequence of adjacent word tokens (a text substring) which are mutually related through dependency links. The fact that two substrings are assigned different chunks does not necessarily entail that there is no dependency relationship linking the two. Simply, this means that, on the basis of the available lexical knowledge, it is impossible to state unambiguously what chunk relates to its neighbouring chunks and what the nature of this relationship is.

#### 4.3.1.4 Assignment

Four different attributes are needed for the description of chunks:

- **id**: the unique identifier
- **href**: a sequence of `<mw>` identifiers
- **type**: the chunk category
- **broken**: whether the chunk is complete (default value) or interrupted

**Type**
Each identified chunk must be labelled as to its category. Chunk categories, to be expressed as values of the `type` attribute, are given in the table below:

| NAME | TYPE |
|---|---|
| ADJ | adjectival chunk |
| ADV | adverbial chunk |
| FV | finite verb chunk |
| G | gerundival chunk |
| I | infinitival chunk |
| N | nominal chunk |
| P | prepositional chunk |
| PART | participial chunk |
| U | unassigned |

In the following we provide basic guidelines for the identification of chunk categories, according to the SPARKLE specifications. Examples are given for Italian and English.

ADJ

Adjectival chunks are chunks beginning with any premodifying adverbs and intensifiers and ending with a head adjective. Adjectival phrases occurring in pre-nominal position are not marked as distinct chunks since their relationship to the governing noun is unambiguously identified within the nominal chunk.

ADJ chunks thus include:

- post-nominal adjectival phrases, either immediately following the noun they modify or placed further down in the sentence:
    - [*un bravo bambino*]$_{N}$
    - [*un bambino*]$_{N}$ [*bravo*]$_{ADJ}$

      [‘a good boy’]$_{N}$
    - [*la progettazione*]$_{N}$ [*di tecniche*]$_{P}$ [*di base*]$_{P}$ [*indispensabili*]$_{ADJ}$ [*al progresso*]$_{P}$ [*industriale*]$_{ADJ}$

      [‘the design]$_{N}$[of basic techniques]$_{P}$ [indispensable]$_{ADJ}$ [to the industrial progress’]$_{P}$
- predicate adjectival phrases, e.g.
    - [*diventa*]$_{FV}$ [*più difficile*]$_{ADJ}$

      [‘(it)]$_{N}$[gets]$_{FV}$ [more difficult’]$_{ADJ}$
    - [*lo considera*]$_{FV}$ [*molto opportuno*]$_{ADJ}$

      [‘(he)]$_{N}$ [considers]$_{FV}$ [it]$_{N}$ [very appropriate’]$_{ADJ}$

ADV

Adverbial chunks are chunks extending form any adverbial pre-modifier to the head adverb. Adverbial phrases occurring between an auxiliary verb and a past participle are not isolated as distinct chunks due to their unambiguous dependency on the verb. By the same token, adverbs which happen to immediately premodify verbs or adjectives are respectively part of either verbal or adjectival chunk. Noun phrases used adverbially (e.g. *last week*, *this morning*) are treated as nominal chunks. E.g.:

- [*ha sempre camminato*]$_{FV}$ [*molto*]$_{ADV}$

  [‘(he)]$_{N}$ [has always walked]$_{FV}$ [a lot’]$_{N}$
- [*ha finito*]$_{FV}$ [*molto rapidamente*]$_{ADV}$

  [‘(he)]$_{N}$ [has finished]$_{FV}$ [very quickly’]$_{ADV}$

FV

Finite verbal chunks include modals, ordinary and causative auxiliaries as well as medial adverbs and clitic pronouns, and they end at the head verb.

- verbal chunks with auxiliary or modal verb and medial adverb:
    - [*was always caught*]$_{FV}$

- [*could not go*]FV

- verbal chunk with pre-modifying adverb:
  - [*certainly caught*]FV
  - [*non ha mai fatto*]FV [*così*]ADV
    ['(he)]N [has never done]FV [so']ADV
- copulas are treated as main verbs if they are followed by something other than a participle or predicate adjective:
  - [*he*]N [*is*]FV [*a nice man*]N
  - [*this*]N [*is good*] FV
- fronted auxiliaries and modals constitute separate FVs:
  - [*può*]FV [*la commissione*]N [*deliberare*]I [*su questa materia?*]P
    ['can]FV [the commission]N [deliberate]I [on this topic?']P
- predicate adjectives are included in FV chunks:
  - [*is fun*] FV, [*is interesting*] FV
- verb particles are not included in FV chunks:
  - [*John*]N [*certainly screwed*]FV*up* [*that time*]N
- light nouns and pieces of V-N idioms are not included in FV chunks:
  - [*took*]FV [*place*]N
  - [*take*]FV [*advantage*]N [*of it*]P
- periphrastic causative constructions with no intervening nominal chunk:
  - [*fece studiare*]FV [*il bambino*]N
    ['(he)]N [let]FV [the child]N [study']I
- clitic pronouns are part of the chunk headed by the immediately ensuing or preceding verb:
  - [*lo ha sempre fatto*]FV
    ['(he)]N [has always done]FV [it']N
- lexically adjacent but syntactically independent verb chunks should be so marked:
  - [*I*]N [*wished*]FV [*the champagne*]N [*I*]N [*ordered*]FV [*would come*]FV
  - [*Er*]N [*hat*]FV [*Bücher*]N [*die*]N [*er*]N [*schon gelesen hat*]FV [*bestellt*]FV

    ['He]N [ordered]FV [books]N [which]N [he]N [has already read']FV

## G

The "G" value marks gerundival chunks. When part of a tensed group (e.g. in the progressive construction), the gerundival verb form is not marked independently (but rather as part of a FV). G chunks include gerunds functioning as noun phrases but not those functioning as adjectives.

- [*sta studiando*]FV
  ['(he)]N [is studying']FV
- [*studiando*]G [*ho imparato*]FV [*molto*]ADV
  ['by studying]G [I]N [have learned]FV [a lot']N
- [*studying*]G [*geography*] N [*is boring*]FV
- [*Flying planes*]N [*cannot be boarded*]FV
- [*Flying*]G [*planes*]N [*is fun*]FV

## I

Infinitival chunks include both bare infinitives and infinitives introduced by a preposition.

- [*ha promesso*] FV [*di arrivare*] I [*presto*] ADV
  ['(he)] N [has promised] FV [to arrive] I [early'] ADV
- [*desidera*] FV [*partire*] I [*domani*] ADV
  ['(he)] N [wishes] FV [to leave] I [tomorrow] ADV

## N

Noun chunks are chunks extending from the beginning of the noun phrase to the head noun. They include nominal chunks headed by nouns, pronouns, verbs in their infinitival form when preceded by an article (i.e. Italian nominalised infinitival constructions) and proper names. All kinds of modifiers and/or specifiers occurring between the beginning of the noun phrase and the head noun are included in N chunks. E.g.:

- [*un bravo bambino*] N
  ['a good boy'] N
- [*tutte le possibili soluzioni*] N
  ['all the possible solutions'] N
- [*la sua ultima battaglia*] N
  ['his last fight'] N
- [*i sempre più frequenti contatti*] N
  ['the more and more frequent contacts'] N

## P

Prepositional chunks are chunks which extend from the preposition to the head of the embedded noun phrase. Typical instances of P chunks are:

- [*al bravo bambino*] P
  ['to the good boy'] P

- [*per i prossimi due anni*] P
  ['for the next two years'] P
- [*fino a un certo punto*] P
  ['up to a certain point'] P

## PART

A Past participle chunk includes participial constructions such as:

- [*finito*] PART [*il lavoro*] N [*Giovanni*] N [*andò*] FV [*a casa*] P
  ['(having) finished] PART [the job] N, [John] N [went] FV [home'] N

## U

The U value indicates that a given chunk cannot be assigned any other value, in general because it is incomplete due to interruption.

For any language-specific issue the interested reader is referred to Abney (1996), Carroll et al. (1997), Federici, Montemagni and Pirrelli (1996).

### Broken

The attribute `broken` supplies a way for representing chunk partials and discontinuous chunks, which are often encountered as a result of interruptions, retracings, and so on.

### 4.3.1.5 Examples

(1) *Hello, can I help you?*

mword.xml

```
...
<mw id="mw_001">hello</mw>
<mw id="mw_002">can</mw>
<mw id="mw_003">I</mw>
<mw id="mw_004">help</mw>
<mw id="mw_005">you</mw>
...
```

chunk.xml

```
...
<ch id="ch_001" type="ADV" href="mword.xml#id(mw_001)"/>
<ch id="ch_002" type="FV"  href="mword.xml#id(mw_002)"/>
<ch id="ch_003" type="N"   href="mword.xml#id(mw_003)"/>
<ch id="ch_004" type="FV"  href="mword.xml#id(mw_004)"/>
<ch id="ch_005" type="N"   href="mword.xml#id(mw_005)"/>
...
```

(2) *La preghiamo di rispondere alle domande del sistema*
('We ask you to reply to the systems's questions')

mword.xml

```
...
<mw id="mw 008">la</mw>
```

```
<mw id="mw_009">preghiamo</mw>
<mw id="mw_010">di</mw>
<mw id="mw_011">rispondere</mw>
<mw id="mw_012">alle</mw>
<mw id="mw_013">domande</mw>
<mw id="mw_014">del</mw>
<mw id="mw_015">sistema</mw>
...
```

chunk.xml

```
...
<ch id="ch_007" type="FV" href="mword.xml#id(mw_008)..id(mw_009)"/>
<ch id="ch_008" type="I"  href="mword.xml#id(mw_010)..id(mw_011)"/>
<ch id="ch_009" type="P"  href="mword.xml#id(mw_012)..id(mw_013)"/>
<ch id="ch_010" type="P"  href="mword.xml#id(mw_014)..id(mw_015)"/>
...
```

The interrupted chunk *you kn-* can be annotated as follows:

mword.xml

```
...
<mw id="mw_001" type="PD">you</mw>
<mw id="mw_002" type="U">kn-</mw>
...
```

chunk.xml

```
...
<ch id="ch_001" type="N"            href="mword.xml#id(mw_001)">
<ch id="ch_002" type="U" broken="Y" href="mword.xml#id(mw_002)"/>
...
```

#### 4.3.1.6 Markup Table

| <ch> | |
|---|---|
| id | [ASCII] |
| href | <mw> |
| type | ADJ, PA, ADV, SUBORD, N, P, FV, G, I, PART, Di, ADJ_PART, COORD, U |
| broken | Y, N |

### 4.3.2 <potgov>: Potential Governor

#### 4.3.2.1 Description

A *potential governor* is the lexical head of the chunk, that is the lexical element (as opposed to a grammatical element), within a chunk, which neighbouring chunks can syntactically combine with in a dependency relation. Note that the specific nature of this relation plays no role in the definition of "potential governor". In fact, a potential governor can be either the head of a dependency or a dependent itself. Thus, it only represents the lexical hook on which other chunks can lean syntactically.

Grammatical words, such as auxiliaries and prepositions are here represented as chunk-internal elements other than potential governors. This choice is geared towards combining grammatical and lexical information in the most informative and manageable way and pave the way to functional annotation.

#### 4.3.2.2 Data Source

For the tagging of potential governors, the markup of morphological words (<mw>) is necessary.

#### 4.3.2.3 Segmentation/Selection

For each chunk type, a potential governor is always the rightmost element of a chunk. In the table below, chunks are classified according to the type of head they require.

| CHUNK TYPE | POSSIBLE HEADS |
|---|---|
| ADJ | adj |
| ADV | adv |
| N | noun, pron, verb |
| P | noun, pron, verb |
| FV | verb |
| G | verb |
| I | verb |
| PART | verb |

#### 4.3.2.4 Assignment

Two different attributes are needed for the description of potential governors:

- **id**: the unique identifier
- **href**: a sequence of <mw> identifiers

#### 4.3.2.5 Examples

(1) *a boxcar of oranges*

mword.xml

```
...
<mw id="mw_020">a</mw>
<mw id="mw 021">boxcar</mw>
```

```
<mw id="mw_022">of</mw>
<mw id="mw_023">oranges</mw>
...
```

chunk.xml

```
...
<ch id="ch_014" type="N" href="mword.xml#id(mw_020)..id(mw_021)">
  <potgov id="p_014" href="mword.xml#id(mw_021)"/>
</ch>
<ch id="ch_015" type="P" href="mword.xml#id(mw_022)..id(mw_023)">
  <potgov id="p_015" href="mword.xml#id(mw_023)"/>
</ch>
...
```

(2) *La preghiamo di rispondere alle domande del sistema*
('We ask you to reply to the systems's questions')

mword.xml

```
...
<mw id="mw_008">la</mw>
<mw id="mw_009">preghiamo</mw>
<mw id="mw_010">di</mw>
<mw id="mw_011">rispondere</mw>
<mw id="mw_012">alle</mw>
<mw id="mw_013">domande</mw>
<mw id="mw_014">del</mw>
<mw id="mw_015">sistema</mw>
...
```

chunk.xml

```
...
<ch id="ch_007" type="FV" href="mword.xml#id(mw_008)..id(mw_009)"/>
  <potgov id="p_007"      href="mword.xml#id(mw_009)"/>
</ch>
<ch id="ch_008" type="I"  href="mword.xml#id(mw_010)..id(mw_011)"/>
  <potgov id="p_008"      href="mword.xml#id(mw_011)"/>
</ch>
<ch id="ch_009" type="P"  href="mword.xml#id(mw_012)..id(mw_013)"/>
  <potgov id="p_009"      href="mword.xml#id(mw_013)"/>
</ch>
<ch id="ch_010" type="P"  href="mword.xml#id(mw_014)..id(mw_015)"/>
  <potgov id="p_010"      href="mword.xml#id(mw_015)"/>
</ch>
...
```

#### 4.3.2.6 Markup Table

| <potgov> | |
|---|---|
| id | [ASCII] |
| href | <mw> |

### 4.3.3 <aux>: Auxiliary Verb

#### 4.3.3.1 Description

The auxiliary verb in a verbal chunk, e.g. "have" in *I have said*, or "was" in *was always caught*.

#### 4.3.3.2 Data Source

For the tagging of auxiliaries, the markup of morphological words is necessary.

#### 4.3.3.3 Assignment

Two different attributes are needed for the description of auxiliaries:

- **id**: the unique identifier
- **href**: a sequence of <mw> identifiers

#### 4.3.3.4 Examples

(1) *I have said*

mword.xml

```
...
<mw id="mw_001">I</mw>
<mw id="mw_002">have</mw>
<mw id="mw_003">said</mw>
...
```

chunk.xml

```
...
<ch id="ch_001" type="N"  href="mword.xml#id(mw_001)">
  <potgov id="p_001"      href="mword.xml#id(mw_001)"/>
</ch>
<ch id="ch_002" type="FV" href="mword.xml#id(mw_002)..id(mw_003)">
  <potgov id="p_002"      href="mword.xml#id(mw_003)"/>
  <aux id="aux_001"       href="mword.xml#id(mw_002)"/>
</ch>
...
```

(2) *ho prenotato un altro biglietto*
('(I) have booked another ticket')

mword.xml

```
...
<mw id="mw_001">ho</mw>
<mw id="mw_002">prenotato</mw>
<mw id="mw_003">un</mw>
<mw id="mw_004">altro</mw>
<mw id="mw_005">biglietto</mw>
...
```

chunk.xml

```
...
<ch id="ch_001" type="FV" href="mword.xml#id(mw_001)..id(mw_002)">
  <potgov id="p_001"      href="mword.xml#id(mw_002)"/>
  <aux id="aux_001"       href="mword.xml#id(mw_001)"/>
</ch>
<ch id="ch_002" type="N"  href="mword.xml#id(mw_003)..id(mw_005)">
  <potgov id="p_002"      href="mword.xml#id(mw_005)"/>
</ch>
...
```

#### 4.3.3.5 Markup Table

| <aux> | |
|---|---|
| id | [ASCII] |

| href | <mw> |
|---|---|

### 4.3.4 <cop>: Copulas

#### 4.3.4.1 Description

This element marks all forms of ‘be’ functioning as a copula, e.g. "is" in *this is good*, or "è" (‘is’) in *la prenotazione è obbligatoria* ‘reservation is obligatory’ .

#### 4.3.4.2 Data Source

For the tagging of copulas, the markup of morphological words is necessary.

#### 4.3.4.3 Assignment

Two different attributes are needed for the description of copulas:

- **id**: the unique identifier
- **href**: a sequence of <mw> identifiers

#### 4.3.4.4 Examples

(1) *this is good*

mword.xml

```
...
<mw id="mw_001">this</mw>
<mw id="mw_002">is</mw>
<mw id="mw_003">good</mw>
...
```

chunk.xml

```
...
<ch id="ch_001" type="N"  href="mword.xml#id(mw_001)">
  <potgov id="p_001"      href="mword.xml#id(mw_001)"/>
</ch>
<ch id="ch_002" type="FV" href="mword.xml#id(mw_002)..id(mw_003)">
  <potgov id="p_002"      href="mword.xml#id(mw_003)"/>
  <cop id="cop_001"       href="mword.xml#id(mw_002)"/>
</ch>
...
```

(2) *la prenotazione è obbligatoria*
(‘reservation is obligatory’)

mword.xml

```
...
<mw id="mw_001">la</mw>
<mw id="mw_002">prenotazione</mw>
<mw id="mw_003">è</mw>
<mw id="mw_003">obbligatoria</mw>
...
```

chunk.xml

```
...
<ch id="ch_001" type="N"  href="mword.xml#id(mw_001)..id(mw_002)">
```

```
  <potgov id="p_001"      href="mword.xml#id(mw_002)"/>
</ch>
<ch id="ch_002" type="FV" href="mword.xml#id(mw_003)..id(mw_004)">
  <potgov id="p_002"      href="mword.xml#id(mw_004)"/>
  <cop id="cop_001"       href="mword.xml#id(mw_003)"/>
</ch>
...
```

#### 4.3.4.5 Markup Table

| <cop> | |
|---|---|
| id | [ASCII] |
| href | <mw> |

### 4.3.5 <intro>: Introducer

#### 4.3.5.1 Description

An underspecified label for the grammatical unit introducing a prepositional, infinitival or verbal chunk, e.g. "to" in *to the restaurant*, "to" in *I need to wash my hair*, or "of" in *of doing*.

#### 4.3.5.2 Data Source

For the tagging of introducers, the markup of morphological words is necessary.

#### 4.3.5.3 Assignment

Two different attributes are needed for the description of introducers:

- **id**: the unique identifier
- **href**: a sequence of <mw> identifiers

#### 4.3.5.4 Examples

(1) *a boxcar of oranges*

mword.xml

```
...
<mw id="mw_020">a</mw>
<mw id="mw_021">boxcar</mw>
<mw id="mw_022">of</mw>
<mw id="mw_023">oranges</mw>
...
```

chunk.xml

```
...
<ch id="ch_014" type="N" href="mword.xml#id(mw_020)..id(mw_021)">
  <potgov id="p_014"     href="mword.xml#id(mw_021)"/>
</ch>
```

```
<ch id="ch_015" type="P" href="mword.xml#id(mw_022)..id(mw_023)">
  <potgov id="p_015"     href="mword.xml#id(mw_023)"/>
  <intro id="i_007"      href="mword.xml#id(mw_022)"/>
</ch>
...
```

(2) *La preghiamo di rispondere*
(‘we ask you to reply’)

mword.xml

```
...
<mw id="mw_008">la</mw>
<mw id="mw_009">preghiamo</mw>
<mw id="mw_010">di</mw>
<mw id="mw_011">rispondere</mw>
...
```

chunk.xml

```
...
<ch id="ch_007" type="FV" href="mword.xml#id(mw_008)..id(mw_009)">
  <potgov id="p_007"      href="mword.xml#id(mw_009)"/>
</ch>
<ch id="ch_008" type="I"  href="mword.xml#id(mw_010)..id(mw_011)">
  <intro id="i_001"       href="mword.xml#id(mw_010)"/>
  <potgov id="p_008"      href="mword.xml#id(mw_011)"/>
</ch>
...
```

#### 4.3.5.5 Markup Table

| <intro> | |
|---|---|
| id | [ASCII] |
| href | <mw> |

### 4.3.6 <modal>: Modal Auxiliary

#### 4.3.6.1 Description

A label for modal auxiliaries such as "can", "have to", "may", "must", "need", "ought to".

#### 4.3.6.2 Data Source

For the tagging of modal auxiliaries, the markup of morphological words is necessary.

#### 4.3.6.3 Assignment

Two different attributes are needed for the description of modal auxiliaries:

- **id**: the unique identifier
- **href**: a sequence of <mw> identifiers

### 4.3.6.4 Examples

(1) *I must admit it*

mword.xml

```
...
<mw id="mw_001">I</mw>
<mw id="mw_002">must</mw>
<mw id="mw_003">admit</mw>
<mw id="mw_004">it</mw>
...
```

chunk.xml

```
...
<ch id="ch_001" type="N"  href="mword.xml#id(mw_001)">
  <potgov id="p_001"      href="mword.xml#id(mw_001)"/>
</ch>
<ch id="ch_002" type="FV" href="mword.xml#id(mw_002)..id(mw_003)">
  <potgov id="p_002"      href="mword.xml#id(mw_003)"/>
<modal id="mo_001"        href="mword.xml#id(mw_002)"/>
</ch>
<ch id="ch_003" type="N"  href="mword.xml#id(mw_003)">
  <potgov id="p_003"      href="mword.xml#id(mw_003)"/>
</ch>
<ch id="ch_001" type="N"  href="mword.xml#id(mw_001)">
  <potgov id="p_001"      href="mword.xml#id(mw_001)"/>
</ch>
<ch id="ch_002" type="FV" href="mword.xml#id(mw_002)..id(mw_003)">
  <potgov id="p_002"      href="mword.xml#id(mw_003)"/>
  <modal id="mo_001"      href="mword.xml#id(mw_002)"/>
</ch>
<ch id="ch_003" type="N"  href="mword.xml#id(mw_003)">
  <potgov id="p_003"      href="mword.xml#id(mw_003)"/>
</ch>
...
```

(2) *lo devo ammettere*
('I must admit it')

mword.xml

```
...
<mw id="mw_001">lo</mw>
<mw id="mw_002">devo</mw>
<mw id="mw_003">ammettere</mw>
...
```

chunk.xml

```
...
<ch id="ch_001" type="FV" href="mword.xml#id(mw_001)..id(mw_003)">
  <potgov id="p_001"      href="mword.xml#id(mw_003)"/>
  <modal id="mo_001"      href="mword.xml#id(mw_002)"/>
</ch>
...
```

### 4.3.6.5 Markup Table

| <modal> | |
|---|---|
| id | [ASCII] |
| href | <mw> |

### 4.3.7 <caus>: Causative Verbs

#### 4.3.7.1 Description

A label for verbs such as ‘let’, ‘make’ and ‘cause’ functioning in Italian, French and Spanish causative constructions.

#### 4.3.7.2 Data Source

For the tagging of causative verbs, the markup of morphological words is necessary.

#### 4.3.7.3 Assignment

Two different attributes are needed for the description of causative verbs:

- **id**: the unique identifier
- **href**: a sequence of `<mw>` identifiers

#### 4.3.7.4 Examples

(1) *L’ho fatto piangere*
(‘I made him cry’)

mword.xml

```
...
<mw id="mw_001">lo</mw>
<mw id="mw_002">ho</mw>
<mw id="mw_003">fatto</mw>
<mw id="mw_004">piangere</mw>
...
```

chunk.xml

```
...
<ch id="ch_001" type="FV" href="mword.xml#id(mw_001)..id(mw_004)">
  <potgov id="p_001"      href="mword.xml#id(mw_004)"/>
  <aux id="aux_001"       href="mword.xml#id(mw_002)"/>
  <caus id="caus_001"     href="mword.xml#id(mw_003)"/>
</ch>
...
```

#### 4.3.7.5 Markup Table

| <caus> | |
|---|---|
| id | [ASCII] |
| href | <mw> |

**Annotation at the chunking level**

Hereafter we provide an example of annotation at the chunking level.

chunk.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE chunk_file SYSTEM "chunk.dtd">
<chunk_file>
<ch id="ch_001" type="FV"   href="mword#id(mw_001)">
  <potgov id="p_001"        href="mword.xml#id(mw_001)"/>
</ch>
<ch id="ch_002" type="N"    href="mword.xml#id(mw_002)..id(mw_003)">
  <potgov id="p_002"        href="mword.xml#id(mw_003)"/>
</ch>
<ch id="ch_003" type="I"    href="mword.xml#id(mw_004)..id(mw_005)">
  <potgov id="p_003"        href="mword.xml#id(mw_005)"/>
</ch>
<ch id="ch_004" type="N"    href="mword.xml#id(mw_006)..id(mw_007)">
  <potgov id="p_004"        href="mword.xml#id(mw_007)"/>
</ch>
<ch id="ch_005" type="P"    href="mword.xml#id(mw_008)..id(mw_009)">
  <potgov id="p_005"        href="mword.xml#id(mw_009)"/>
</ch>
<ch id="ch_006" type="P"    href="mword.xml#id(mw_010)..id(mw_011)">
  <potgov id="p_006"        href="mword.xml#id(mw_011)"/>
</ch>
<ch id="ch_007" type="ADV"  href="mword.xml#id(mw_012)">
  <potgov id="p_007"        href="mword.xml#id(mw_012)"/>
</ch>
<ch id="ch_008" type="ADV"  href="mword.xml#id(mw_013)">
  <potgov id="p_008"        href="mword.xml#id(mw_013)"/>
</ch>
<ch id="ch_009" type="FV"   href="mword.xml#id(mw_014)">
  <potgov id="p_009         href="mword.xml#id(mw_014)"/>
</ch>
<ch id="ch_010" type="N"    href="mword.xml#id(mw_015)..id(mw_016)">
  <potgov id="p_010"        href="mword.xml#id(mw_016)"/>
</ch>
<ch id="ch_011" type="PART" href="mword.xml#id(mw_017)">
  <potgov id="p_011"        href="mword.xml#id(mw_017)"/>
</ch>
<ch id="ch_012" type="N"    href="mword.xml#id(mw_018)">
  <potgov id="p_012"        href="mword.xml#id(mw_018)"/>
</ch>
<ch id="ch_013" type="FV"   href="mword.xml#id(mw_019)..id(mw_021)">
  <potgov id="p_013"        href="mword.xml#id(mw_021)"/>
  <aux id="aux_001"         href="mword.xml#id(mw_019)"/>
</ch>
<ch id="ch_014" type="N"    href="mword.xml#id(mw_022)..id(mw_023)">
  <potgov id="p_014"        href="mword.xml#id(mw_023)"/>
</ch>
<ch id="ch_015" type="P"    href="mword.xml#id(mw_024)..id(mw_026)">
  <potgov id="p_015"        href="mword.xml#id(mw_026)"/>
  <intro id="i_0o1"         href="mword.xml#id(mw_024)"/>
</ch>
<ch id="ch_016" type="PART" href="mword.xml#id(mw_027)">
  <potgov id="p_016"        href="mword.xml#id(mw_027)"/>
</ch>
<ch id="ch_017" type="ADV"  href="mword.xml#id(mw_028)">
  <potgov id="p_017"        href="mword.xml#id(mw_028)"/>
</ch>
<ch id="ch_018" type="FV"   href="mword.xml#id(mw_029)">
  <potgov id="p_018"        href="mword.xml#id(mw_029)"/>
</ch>
<ch id="ch_019" type="N"    href="mword.xml#id(mw_030)..id(mw_031)">
  <potgov id="p_019"        href="mword.xml#id(mw_031)"/>
</ch>
<ch id="ch_020" type="ADV"  href="mword.xml#id(mw_032)">
  <potgov id="p_020"        href="mword.xml#id(mw_032)"/>
</ch>
<ch id="ch_021" type="FV"   href="mword.xml#id(mw_033)">
  <potgov id="p_021"        href="mword.xml#id(mw_033)"/>
</ch>
<ch id="ch_022" type="N"    href="mword.xml#id(mw_034)..id(mw_035)">
  <potgov id="p_022"        href="mword.xml#id(mw_035)"/>
</ch>
<ch id="ch_023" type="P"    href="mword.xml#id(mw_036)..id(mw_037)">
  <potgov id="p_023"        href="mword.xml#id(mw_037)"/>
<intro id="i_002"           href="mword.xml#id(mw_036)"/>
</ch>
<ch id="ch_024" type="P"    href="mword.xml#id(mw_038)..id(mw_039)">
  <potgov id="p_024"        href="mword.xml#id(mw_039)"/>
<intro id="i_003"           href="mword.xml#id(mw_038)"/>
</ch>
<ch id="ch_025" type="ADV"  href="mword.xml#id(mw_040)">
  <potgov id="p_025"        href="mword.xml#id(mw_040)"/>
</ch>
```

```
<ch id="ch_026" type="FV"   href="mword.xml#id(mw_041)">
  <potgov id="p_026"        href="mword.xml#id(mw_041)"/>
</ch>
<ch id="ch_027" type="N"    href="mword.xml#id(mw_042)..id(mw_044)">
  <potgov id="p_027"        href="mword.xml#id(mw_044)"/>
</ch>
<ch id="ch_028" type="ADV"  href="mword.xml#id(mw_045)">
  <potgov id="p_0248"       href="mword.xml#id(mw_045)"/>
</ch>
<ch id="ch_029" type="FV"   href="mword.xml#id(mw_046)">
  <potgov id="p_029"        href="mword.xml#id(mw_046)"/>
</ch>
<ch id="ch_030" type="N"    href="mword.xml#id(mw_047)..id(mw_048)">
  <potgov id="p_030"        href="mword.xml#id(mw_048)"/>
</ch>
<ch id="ch_031" type="P"    href="mword.xml#id(mw_049)..id(mw_050)">
  <potgov id="p_031"        href="mword.xml#id(mw_050)"/>
  <intro id="i_004"         href="mword.xml#id(mw_049)"/>
</ch>
<ch id="ch_032" type="P"    href="mword.xml#id(mw_051)..id(mw_052)">
  <potgov id="p_032"        href="mword.xml#id(mw_052)"/>
  <intro id="i_005"         href="mword.xml#id(mw_051)"/>
</ch>
<ch id="ch_033" type="N"    href="mword.xml#id(mw_054)">
  <potgov id="p_033"        href="mword.xml#id(mw_054)"/>
</ch>
<ch id="ch_034" type="P"    href="mword.xml#id(mw_055)..id(mw_056)">
  <potgov id="p_034"        href="mword.xml#id(mw_056)"/>
  <intro id="i_006"         href="mword.xml#id(mw_055)"/>
</ch>
</chunk_file>
```

# 5 Functional Annotation Coding Module

## 5.1 Coding Purpose

This section is concerned with syntactic annotation at the functional level.

The functional analysis of a sentence such as *John gave the book to Mary* provides information about how grammatical relations such as *subject*, *object* and *indirect object* of the verb *give* are instantiated in context. In particular, we would like to say that *John* is the subject, *book* the object and *Mary* the indirect object of *give*. We thus intend a functional relation as a binary relation between a verb and the lexical head of a phrase syntactically depending on the verb, according to the following schema:

dep_type (lex_head.<head_features>, dependent.<dep_features>)

For example:

subj(give <diath="active", tense="past">,John <case="nominative">)

dobj(give<diath="active", tense="past">,book)

iobj(give,Mary <intro="to">)

To be more concrete, a binary functional relationship can be represented formally as consisting of the following three types of information:

- **i.** *type of relationship* involved: example, the functional relation of the pair (give, Mary) in the sentence *John gave the book to Mary* is *indirect object*
- **ii.** the *terms* of the relationship (i.e. the linguistic units in text which enter a given functional relationship): (give, Mary)
- **iii.** the *syntactic category of the dependent* and possibly *case features* and other *morphosyntactic features*: example, the dependent in the pair (give, Mary) has category "non-clausal" (n-c); given the phrase *America's production*, the dependent in the pair (production, America) has category "non-clausal" and is realised in genitive case.

## 5.2 Existing and selected schemes

The scheme presented here is based on SPARKLE specifications (Carroll et al., 1997), but departs from it at two levels. At the level of information to be encoded obligatorily, the types of functional relation listed here represent only a subset of the types recommended in the SPARKLE specifications, since all language specific tags are not considered in the MATE list. Moreover, in some cases, MATE recommended functional relations correspond to more than one functional relation in SPARKLE. This because SPARKLE uses (possibly underspecified) atomic functions, while MATE makes provision for functional information to be represented in a distributed way, through use of more than one attribute-value pair.

At the level of possible extensions of the notational core foreseen in SPARKLE, we further suggest annotation of strictly non functional links, such as those holding between conjuncts, or the binding relationship defined between a relative pronoun and the lexical head modified by the pronoun.

Other minor differences between the two schemes, such as the one concerning the treatment of omitted subjects in pro-drop languages, will be dealt with wherever appropriate.

## 5.3 Markup Declaration

Encoding is carried out by means of <funct> elements, which point to lexical tokens only indirectly.

The terms of the relationship are annotated through two dedicated elements, <head> and <dep>, which are hierarchically embedded within <funct> elements and point to the relevant lexical tokens in the resource file.

The type of relationship involved (see item i. in the list above) is represented by means of a list of values for the attribute type, further specifying <dep> elements.

Morphosyntactic features can be specified, when needed, through attributes in <head> and <dep> elements.

<head> attributes are: diath (i.e., the diathesis of a verbal head, whether active, passive, or middle), tense, person, number and gender, respectively the morphosyntactic tense, person, number and gender of the head.

<dep> attributes are: intro (for introducer, i.e. the element which possibly introduces the dependent), case (i.e., the case of the dependent), and synt_real (i.e., the particular syntactic realization of the dependent, whether clausal or non clausal).

Elements for this level of annotation are recapitulated below:

```
<funct>
     <head>
     <dep>
```

### 5.3.1 <funct>

#### 5.3.1.1 Description

A <funct> element contains one or more <dep> elements, representing all dependents of the same head, plus one <head> element, specifying the head itself. Different types of functional relations are specified as different values of the attribute type of the element <dep>, as detailed below.

#### 5.3.1.2 Data Source

Annotation of <funct> presupposes the morphological markup of <head> and <dep> elements (see below).

#### 5.3.1.3 Selection

A functional relation classically expresses the syntactic dependency between two lexical heads, one of which is a verb. This definition can be stretched so as to include other lexical dependency

relations such as those between - say - a modifier and a noun (e.g., in *the destruction of the city*, where *city* is understood as a modifier of *destruction*), or an adjective and its infinitival complement (e.g., in *keen to please*). It is important to emphasise here that neither definition includes annotation of dependency relations involving a lexical and a grammatical head, as in the case of the syntactic relation between a verb and its auxiliary (e.g., in *has tried*), or a determiner and its ensuing noun (e.g., in *the dog*). To give a practical example, consider the following sentence:

*Paul said that he will accept Microsoft's offer*

If only verb heads are considered, then the mark-up of functional relations will select the following elements:

subj(say,Paul)
comp(say, accept)
subj(accept,he)
dobj(accept,offer)

If also heads other than verbal are considered, then annotation will include the following piece of information:

mod(offer, Microsoft)

All these functions will be represented in XML as values of the attribute type in the `<dep>` elements.

#### 5.3.1.4 Assignment

One attribute is needed for the description of the element <`funct`>:

- **id**: a unique identifier

#### 5.3.1.5 Examples

(1) *John arrived in Paris*

mword.xml

```
...
<mw id="mw_001">John</mw>
<mw id="mw_002">arrived</mw>
<mw id="mw_003">in</mw>
<mw id="mw_004">Paris</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001"            href="mword.xml#id(mw_002)"/>
  <dep id="d_001" type="subj" href="mword.xml#id(mw_001)"/>
  <dep id="d_002" type="iobj" href="mword.xml#id(mw_004)"/>
</funct>
...
```

(2) *John is interested in Mathematics*

mword.xml

```
...
<mw id="mw_001">John</mw>
<mw id="mw_002">is</mw>
<mw id="mw_003">interested</mw>
<mw id="mw_004">in</mw>
<mw id="mw_005">Mathematics</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001"            href="mword.xml#id(mw_002)"/>
  <dep id="d_001" type="subj" href="mword.xml#id(mw_001)"/>
  <dep id="d_002" type="comp" href="mword.xml#id(mw_003)"/>
</funct>
<funct id="funct_002">
  <head id="h_002"            href="mword.xml#id(mw_003)"/>
  <dep id="d_003" type="mod"  href="mword.xml#id(mw_005)"/>
</funct>
...
```

(3) *Paul said that he will accept the job*

mword.xml

```
...
<mw id="mw_001">Paul</mw>
<mw id="mw_002">said</mw>
<mw id="mw_003">that</mw>
<mw id="mw_004">he</mw>
<mw id="mw_005">will</mw>
<mw id="mw_006">accept</mw>
<mw id="mw_007">the</mw>
<mw id="mw_008">job</mw>
...
```

funct.xml

```
...
<funct id="funct_001" >
  <head id="h_001"            href="mword.xml#id(mw_002)"/>
  <dep id="d_001" type="subj" href="mword.xml#id(mw_001)"/>
  <dep id="d_002" type="comp" href="mword.xml#id(mw_006)"/>
</funct>
<funct id="funct_002">
  <head id="h_002"            href="mword.xml#id(mw_006)"/>
  <dep id="d_003" type="subj" href="mword.xml#id(mw_004)"/>
  <dep id="d_004" type="dobj" href="mword.xml#id(mw_008)"/>
</funct>
...
```

(4) *Paul intends to leave IBM*

mword.xml

```
...
<mw id="mw_001">Paul</mw>
<mw id="mw_002">intends</mw>
<mw id="mw_003">to</mw>
<mw id="mw_004">leave</mw>
<mw id="mw_005">IBM</mw>
...
```

funct.xml

```
...
<funct id="funct_001" >
  <head id="h_001"            href="mword.xml#id(mw_002)"/>
  <dep id="d_001" type="subj" href="mword.xml#id(mw_001)"/>
  <dep id="d_002" type="comp" href="mword.xml#id(mw_004)"/>
</funct>
<funct id="funct_002">
  <head id="h_002"            href="mword.xml#id(mw_004)"/>
```

```
  <dep id="d_003" type="subj" href="mword.xml#id(mw_001)"/>
  <dep id="d_004" type="dobj" href="mword.xml#id(mw_005)"/>
</funct>
...
```

(5) *he ate the cake without asking*

mword.xml

```
...
<mw id="mw_001">he</mw>
<mw id="mw_002">ate</mw>
<mw id="mw_003">the</mw>
<mw id="mw_004">cake</mw>
<mw id="mw_005">without</mw>
<mw id="mw_006">asking</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001"            href="mword.xml#id(mw_002)"/>
  <dep id="d_001" type="subj" href="mword.xml#id(mw_001)"/>
  <dep id="d_002" type="dobj" href="mword.xml#id(mw_004)"/>
  <dep id="d_003" type="mod"  href="mword.xml#id(mw_006)"/>
</funct>
...
```

#### 5.3.1.6 Markup Table

| <funct> | |
|---|---|
| id | [ASCII] |

### 5.3.2 <head>: Lexical Heads

#### 5.3.2.1 Description

A <head> element is used to mark the lexical head in a functional relation.

#### 5.3.2.2 Data Source

Tagging of lexical heads presupposes the morphological markup of words.

#### 5.3.2.3 Segmentation/Selection

Classically, the head of a functional relation is a verb. However, we also make provision for the assignment of functional relations headed by nouns and adjectives, as in *the fact that John won* or *I'm ready to leave*. As already emphasised above, a head is always lexical. In other words, grammatical words such as auxiliaries, determiners or prepositions do not represent a head in our sense.

#### 5.3.2.4 Assignment

Two attributes are needed for the description of heads:

- **id**: a unique identifier
- **href**: one or more `<mw>` identifiers

The following attributes can additionally be specified:

- **diath**: specifies the diathesis of a verbal head. Possible values are: *active*, *passive*, *middle*
- **tense:** specifies the tense of a verbal head
- **person**: specifies the person of a head
- **number**: specifies the number of a head
- **gender**: specifies the gender of a head

#### 5.3.2.5 Examples

(1) *Paul was employed by Microsoft*

mword.xml

```
...
<mw id="mw_001">Paul</mw>
<mw id="mw_002">was</mw>
<mw id="mw_003">employed</mw>
<mw id="mw_004">by</mw>
<mw id="mw_005">Microsoft</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001" tense="past" diath="passive" href="mword.xml#id(mw_003)"/>
  <dep id="d_001" type="subj"                   href="mword.xml#id(mw_001)"/>
  <dep id="d_002" type="mod"                    href="mword.xml#id(mw_005)"/>
</funct>
...
```

(2) *Microsoft employed Paul*

mword.xml

```
...
<mw id="mw_001">Microsoft</mw>
<mw id="mw_002">employed</mw>
<mw id="mw_003">Paul</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001" diath="active" href="mword.xml#id(mw_002)"/>
  <dep id="d_001" type="subj"     href="mword.xml#id(mw_001)"/>
  <dep id="d_002" type="dobj"     href="mword.xml#id(mw_003)"/>
</funct>
...
```

(3) *he eats a pizza*

mword.xml

```
...
<mw id="mw_001">he</mw>
<mw id="mw_002">eats</mw>
<mw id="mw_003">a</mw>
<mw id="mw_003">pizza</mw>
...
```

funct.xml

```
...
<funct id="funct_001" >
   <head id="h_001" diath="active" person="3" number="sg" href="mword.xml#id(mw_002)"/>
   <dep id="d_001" type="subj"                            href="mword.xml#id(mw_001)"/>
   <dep id="d_002" type="dobj"                            href="mword.xml#id(mw_004)"/>
</funct>
...
```

(4) *Maria è arrivata* (‘Maria has arrived’)

mword.xml

```
...
<mw id="mw_001">Maria</mw>
<mw id="mw_002">è</mw>
<mw id="mw_003">arrivata</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001" tense="present perfect" diath="active" person="3"
                   number="sg" gender="f" href="mword.xml#id(mw_003)"/>
  <dep id="d_001" type="subj"             href="mword.xml#id(mw_001)"/>
</funct>
...
```

### 5.3.2.6 Markup Table

| <head> | |
|---|---|
| id | [ASCII] |
| href | <mw> |
| diath | active, passive, middle |
| tense | [ASCII] |
| person | 1, 2, 3 |
| number | sg, pl |
| gender | m, f, n |

## 5.3.3 <dep>: Dependents

#### 5.3.3.1 Description

A <dep> element is used to mark the dependent in a functional relation.

#### 5.3.3.2 Data Source

Tagging of dependents presupposes the morphological markup of words.

#### 5.3.3.3 Segmentation/Selection

As already pointed out above, a functional relationship can involve lexical elements only, whether heads or dependents. This means that grammatical words such as auxiliaries, determiners, conjunctions and prepositions cannot play the role of a dependent in our sense. If needed, lexical information about the type of conjunction or preposition accompanying the lexical dependent in the text is provided through use of the attribute intro (see following section).

#### 5.3.3.4 Assignment

The following attributes are used for the description of dependents:

- **id**: the unique identifier
- **href**: one or more <mw> identifiers
- **type**: the type of functional relation

**Type**
Each identified <dep> element must be labelled as to its type, representing the particular functional relation which can be identified between the dependent and its verbal or nominal head. Types of functional relation, to be expressed as values of the type attribute, are given in the table below:

| MATE function | TYPE | SPARKLE function |
|---|---|---|
| SUBJ | a surface subject relation | SUBJ (overt) |
| DOBJ | a direct object relation | DOBJ |
| OBJ2 | a second non-clausal complement in ditransitive constructions | OBJ2 |
| IOBJ | an indirect object relation | IOBJ |
| MOD | a modifying relation | MOD |
| COMP | the relation between a verb and a clausal or predicative complement | XCOMP, CCOMP |
| U | an underspecified relation | - |

In the following we provide basic guidelines for the identification of functional relations, mainly reflecting SPARKLE specifications.

**SUBJ**

The relation between a verb and its subject:

| | |
|---|---|
| subj(arrive,John) | *John arrived in Paris* |
| subj(employ,Microsoft) | *Microsoft employed 10 C programmers* |
| subj(employ,Paul) | *Paul was employed by Microsoft* |

"subj" refers to the superficial subject of the verb, regardless of it being used in the active or passive voice. Moreover, it can also be used to mark subject control relations and possibly raising to object/subject phenomena, as exemplified below:

| | |
|---|---|
| subj(leave,John) | *John promised Mary to leave* |
| subj(leave,Mary) | *John ordered Mary to leave* |
| subj(leave,Mary) | *Mary was ordered to leave* |
| subj(be,her) | *John believes her to be intelligent* |
| subj(be,John) | *John seems to be intelligent* |

Also clausal subjects are marked through "subj":

| | |
|---|---|
| subj(mean,leave) | *that Nellie left without saying good-bye meant she was still angry* |
| subj(require,win) | *to win the America's Cup requires heaps of cash* |

With pro-drop languages such as Italian, when the subject is not overtly realised the annotation is partial, as specified below:

| | |
|---|---|
| subj(arrivare,_) | *arrivai in ritardo '(I) arrived late'* |

where i) the dependent slot is left unspecified and ii) the morphosyntactic features, which indicate person, number and gender of the subject, can be recovered from the inflectional features associated with the head (see further below).

**DOBJ**

The relation between a verb and its non-clausal direct object, e.g.:

| | |
|---|---|
| dobj(read,book) | *John read many books* |

We suggest that also non passivizable bare NPs following measure verbs (as in *it weighs two stones*) be annotated as direct objects.

**OBJ2**

The relation between a verb and the second non-clausal complement in ditransitive constructions, e.g.:

| | |
|---|---|
| obj2(give,present) | *give Mary a present* |
| obj2(mail,contract) | *mail Mark the contract* |

**IOBJ**

The relation between a verb and a non-clausal complement introduced by a preposition, e.g.:

| | |
|---|---|
| iobj(speak,Mary) | *John speaks to Mary* |
| iobj(give,Mary) | *John gave Mary the contract* |
| iobj(give,Mary) | *John gave the contract to Mary* |
| iobj(live,Rome) | *John lives in Rome* |
| iobj(arrive,airport) | *John arrived at the airport* |
| iobj(inform,arrival) | *John informed Mary of his late arrival* |

**COMP**

The relationship between a verb and a clausal or predicative complement, e.g.:

| | |
|---|---|
| comp(say,accept) | *Paul said that he will accept Microsoft's offer* |
| comp(inform,arrive) | *Paul informed Mary that he would arrive soon* |

| | |
|---|---|
| comp(intend,leave) | *Paul intends to leave IBM* |
| comp(prevent,take) | *This drug prevents you from taking the flu* |
| comp(be,easy) | *Swimming is easy* |
| comp(be,Paris) | *Mary is in Paris* |
| comp(be,manager) | *Paul is the manager* |

**MOD**

The relation "mod" holds between a head and its modifier, both clausal and non-clausal; e.g.

| | |
|---|---|
| mod(walk,slowly) | w*alk slowly* |
| mod(walk,John) | *Walk with John* |
| mod(Picasso,painter) | *Picasso the painter* |
| mod(walk,talk) | *walk while talking* |
| mod(eat,be) | *he ate the cake because he was hungry* |
| mod(eat,ask) | *he ate the cake without asking* |
| mod(flag,red) | *a red flag* |

**mod** is also used:

- to encode the relation between an event noun (including deverbal nouns) and its semantic arguments; e.g.

| | |
|---|---|
| mod(gift,book) | *the gift of a book* |
| mod(gift,Peter) | *the gift of a book by Peter* |
| mod(examination,patient) | *the examination of the patient* |
| mod(examination, doctor) | *the doctor's examination of the patient* |

- to encode the relation between a head and a semantic argument which is syntactically realised as a modifier; thus a *by*-phrase can be analysed as a 'thematically bound adjunct' e.g.

| | |
|---|---|
| mod(kill,Brutus) | *killed by Brutus* |

**U**

In a number of borderline cases the distinction between a modifier and a true argument of a verb is difficult to draw in practice. We observe that this distinction is mainly lexical, as it chiefly depends on the type of information presupposed in the lexical entry of the verb to be annotated, information which, in its turn, may depend on the type of domain which the lexicon is designed to cover. Thus, we suggest that this issue cannot be dealt with once and for all on a principled basis. It is important that any annotation scheme at this level make provision for the possibility of underspecifying the difference between arguments and modifiers by using a function label which subsumes both categories. In our case, this can be achieved by assigning the value U (mnemonic for underspecified) to the `type` attribute. For example:

| | |
|---|---|
| u(post,Eton) | *I posted the letter from Eton* |

The following attributes can be additionally specified:

- **intro**: the element which possibly introduce the dependent in a given functional relation, e.g. prepositions, conjunctions, etc.
- **case**: the case of the dependent
- **synt_real**: it refers to a broad classification of the syntactic realization of a given dependent, with respect to its being clausal or non-clausal, or with respect to the type of clausal/predicative structure (i.e. whether it is an open predicate or a saturated predicate). Possible values of this attribute are clausal (cl) and non clausal (n_cl); clausal dependents can be further specified as open (x) or closed (c). Criteria for value assignment are illustrated below:
  - **x** – a subcategorized argument or modifier containing an empty argument position which must be controlled by a constituent outside it:

| | |
|---|---|
| comp(decide,leave.<synt_real=x>) | *John decided to leave* |
| mod(attend,arrive.<synt_real=x>) | *Having arrived early, we attended the meeting* |
| subj(require,win.<synt_real=x>) | *To win the America's Cup requires heap of cash* |

- **c** – a subcategorized argument or modifier which requires no control by a constituent outside it:

| | |
|---|---|
| comp(say, leave.<synt_real=c>) | *John said he would leave* |
| comp(inform, leave.<synt_real=c>) | *John informed Mary that he would leave* |
| mod(eat, hungry.<synt_real=c>) | *He ate the cake because he was hungry* |
| subj(mean, leave.<synt_real=c>) | *That Nellie left means that she was angry* |

- **nc** – a non-clausal argument or modifier:

| | |
|---|---|
| dobj(eat, pizza.<synt_real=n_cl>) | *John ate a pizza* |
| subj(eat, John<synt_real=n_cl>) | *John ate a pizza* |
| mod(eat, fork<synt_real=n_cl>) | *John ate a pizza with a fork* |
| comp(be, intelligent.<synt_real=n_cl>) | *John is intelligent* |

### 5.3.3.5 Examples

(1) *give to Mary*

mword.xml

```
...
<mw id="mw_001">give</mw>
<mw id="mw_002">to</mw>
<mw id="mw_003">Mary</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001"                      href="mword.xml#id(mw_002)"/>
  <dep id="d_001" type="iobj" intro="to" href="mword.xml#id(mw_003)"/>
</funct>
...
```

(2) *Paul said that he will accept Microsoft's offer*

mword.xml

```
...
<mw id="mw_001">Paul</mw>
<mw id="mw_002">said</mw>
<mw id="mw_003">that</mw>
<mw id="mw_004">he</mw>
<mw id="mw_005">will</mw>
<mw id="mw_006">accept</mw>
<mw id="mw_007">Microsoft's</mw>
<mw id="mw_008">offer</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
   <head id="h_001"                          href="mword.xml#id(mw_002)"/>
   <dep id="d_001" type="comp" intro="that" synt_real="c"
                                             href="mword.xml#id(mw_006)"/>
</funct>
...
```

(3) *Paul intends to leave IBM*

mword.xml

```
...
```

```
<mw id="mw_001">Paul</mw>
<mw id="mw_002">intends</mw>
<mw id="mw_003">to</mw>
<mw id="mw_004">leave</mw>
<mw id="mw_005">IBM</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001"                        href="mword.xml#id(mw_002)"/>
  <dep id="d_001" type="comp" intro="to" synt_real="x"
                                          href="mword.xml#id(mw_004)"/>
</funct>
...
```

(4) *to win the America's Cup requires money*

mword.xml

```
...
<mw id="mw_001">to</mw>
<mw id="mw_002">win</mw>
<mw id="mw_003">the</mw>
<mw id="mw_004">America's</mw>
<mw id="mw_005">Cup</mw>
<mw id="mw_006">requires</mw>
<mw id="mw_007">money</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001" diath="active"                      href="mword.xml#id(mw_006)"/>
  <dep id="d_001" type="subj" intro="to" synt_real="x" href="mword.xml#id(mw_002)"/>
  <dep id="d_002" type="dobj"                          href="mword.xml#id(mw_007)"/>
</funct>
<funct id="funct_002">
  <head id="h_002"                                     href="mword.xml#id(mw_002)"/>
  <dep id="d_003" type="dobj" intro="to" synt_real="x" href="mword.xml#id(mw_004)..id(mw_005)"/>
</funct>
...
```

(5) *dargli*

mword.xml

```
...
<mw id="mw_001">dare</mw>
<mw id="mw_002">gli</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
   <head id="h_001"                       href="mword.xml#id(mw_001)"/>
   <dep id="d_001" type="iobj" case="dat" href="mword.xml#id(mw_002)"/>
</funct>
...
```

### 5.3.3.6 Markup Table

| <dep> | |
|---|---|
| id | [ASCII] |
| href | <mw> |

| | |
|---|---|
| type | subj, dobj, obj2, iobj, mod, comp |
| case | [ASCII] |
| synt_real | n_cl, cl, c, x |
| intro | [ASCII] |

## 5.4 Language-specific constructions

In this section, some language-specific constructions which require some *ad hoc* pieces of annotation are discussed and possible solutions are put forward and motivated.

### 5.4.1 Pro-drop

As specified above (see sec. 5.3.1.4), in pro-drop languages such as Italian or Spanish annotation of subject relation is partial, as illustrated below for the sentence *arrivai in ritardo* '(I) arrived late':

subj(arrivare.<person=1,number=sing>,_)

where i) the dependent slot is left empty and ii) the morphosyntactic features, which indicate person, number (and gender) of the subject, can be recovered from the inflectional features associated with the head.

As an alternative to this annotation, in SPARKLE implicit subjects are annotated as a relation between the verbal head and an empty element *PRO* as follows:

subj(arrivare,pro)

where PRO elements are treated as dummy elements marked at least for person and number features. We believe that this solution should be resisted, however, since it falsely imply that the omitted subject is implicitly realised at some level (the word level?) and that the verb agrees with it.

#### 5.4.1.1 Markup

We represent annotation of subject relations in pro-drop languages as a `<funct>` element which includes one `<head>` element and one `<dep>` element, marked as "subj" for its `type` attribute, with no value for `href`. Agreement features are expected to be recovered from the `<mw>` unit pointed to in the `href` value of the `<head>` element.

(1) *arrivai in ritardo* ('I arrived late')

mword.xml

```
...
<mw id="mw_001">arrivai</mw>
<mw id="mw_002">in</mw>
<mw id="mw_003">ritardo</mw>
...
```

funct.xml

```
...
```

```
<funct id="funct_001">
  <head id="h_001" href="mword.xml#id(mw_001)"/>
  <dep id="d_001" type="subj"/>
</funct>
...
```

### 5.4.2 Impersonal constructions

The representation proposed above for pro-drop phenomena can also be adopted to annotate impersonal constructions (i.e. impersonal verbs or impersonal passives), where the verb form is third person singular and no referential subject is neither understood nor expressed:

*oggi piove* (‘today (it) rains’)

subj(piovere.<v_type=impers,person=3,number=sg,_)
mod(piovere,oggi)

#### 5.4.2.1 Markup

For the representation of impersonal constructions, the set of attributes of `<head>` elements must be extended so as to include a new optional attribute `v_type`, bearing the unique value "impers".

| <head> | |
|---|---|
| v_type | impers |

(1) *oggi piove* (‘today (it) rains)

mword.xml

```
...
<mw id="mw_001">oggi</mw>
<mw id="mw_002">piove</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001" v_type="impers" person="3" href="mword.xml#id(mw_002)"/>
  <dep id="d_001"  type="subj"/>
  <dep id="d_002"  type="mod"                 href="mword.xml#id(mw_001)"/>
</funct>
...
```

### 5.4.2 Expletive subjects

Expletive subjects are referentially empty syntactic subjects, overtly realized by expletive pronouns such as English *it* and *there*. We suggest that they should be annotated as follows:

- *it bothers John that Mary left*

  subj(bother,it)
  dobj(bother,John)
  comp(bother,leave.<introducer="that",synt_real="c">)
  subj(leave,Mary)

- *there is a wasp in the garden*

  subj(be,there)
  comp(be,wasp)
  mod(be,garden.<intro="in",synt_real="nc">)

#### 5.4.2.1 Markup

(1) *it bothers John that Mary left*

mword.xml

```
...
<mw id="mw_001">it</mw>
<mw id="mw_002">bothers</mw>
<mw id="mw_003">John</mw>
<mw id="mw_004">that</mw>
<mw id="mw_005">Mary</mw>
<mw id="mw_006">left</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001"                                        href="mword.xml#id(mw_002)"/>
  <dep id="d_001 type="subj"                              href="mword.xml#id(mw_001)"/>
  <dep id="d_002 type="dobj"                              href="mword.xml#id(mw_003)"/>
  <dep id="d_003 type="comp" intro="that" synt_real="c" href="mword.xml#id(mw_006)"/
</funct>
<funct id="funct_002">
  <head id="h_002"                                        href="mword.xml#id(mw_006)"/>
  <dep id="d_004 type="subj"                              href="mword.xml#id(mw_005)"/>
</funct>
...
```

(2) *there is a wasp in the garden*

mword.xml

```
...
<mw id="mw_001">there</mw>
<mw id="mw_002">is</mw>
<mw id="mw_003">a</mw>
<mw id="mw_004">wasp</mw>
<mw id="mw_005">in</mw>
<mw id="mw_006">the</mw>
<mw id="mw_007">garden</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001"                                        href="mword.xml#id(mw_002)"/>
  <dep id="d_001 type="subj"                              href="mword.xml#id(mw_001)"/>
  <dep id="d_002 type="comp"                              href="mword.xml#id(mw_004)"/>
  <dep id="d_003 type="mod" intro="in" synt_real="nc" href="mword.xml#id(mw_007)"/>
</funct>
<funct id="funct_002">
  <head id="h_002"                                        href="mword.xml#id(mw_006)"/>
  <dep id="d_004 type="subj"                              href="mword.xml#id(mw_005)"/>
</funct>
...
```

### 5.5 Possible Extensions to the basic scheme

In this section we introduce some extensions to the basic scheme described in the previous pages. These extensions may be added on top of the core scheme whenever the user feels the need for

encoding more information, or requires a finer graded categorization of phenomena.

### 5.5.1 Elliptical constructions

We provide here an illustration of how the proposed mark-up scheme could be used for annotation of elliptical constructions such as those found in dialogue texts and gapped conjuncts. Consider, for instance, the following example:

A: *io sono andato a Roma* (‘I went to Rome’)

B: *io a Parigi* (‘Me, to Paris’)

subj(andare,io)
iobj(andare,Roma.<introducer="a">)
subj(ANDARE,io)
iobj(ANDARE,Parigi. <introducer="a">)

The idea here is to link *io* and *a Parigi* in turn B to the verb "andare", which is overtly expressed only in turn A. This can be achieved by annotating an explicit subject relation between *io* and *sono andato* in turn A, and an understood relation of subject holding between *io* and the lemma *ANDARE* in turn B.

#### 5.5.1.1 Markup

The only notable difference concerning annotation of elliptical heads is that, in this case, `<head>` elements no longer point to a morphological word (contained in a "mword.xml" file) in the usual way, but rather refer to the lexical entry corresponding to the elliptical token. This presupposes that a separate file containing all lexical entries be available in XML format.

#### 5.5.1.2 Example

(1) A: *io sono andato a Roma* (‘I went to Rome’)
B: *io a Parigi* (‘Me, to Paris’)

lex.xml

```
...
<lemma id="l_0053" pos="V" orth="andare"/>
<lemma id="l_1321" pos="V" orth="essere"/>
<lemma id="l_0001" pos="P" orth="a"/>
<lemma id="l_2385" pos="V" orth="io"/>
<lemma id="l_5076" pos="N" orth="Roma"/>
<lemma id="l_4503" pos="N" orth="Parigi"/>
...
```

mword.xml

```
...
<mw id="m_001" type="P"> io
  <lexit id="li_001" href="lex.xml#id(l_2385)"/>
</mw>
<mw id="m_002" type="V"> sono
  <lexit id="li_002" href="lex.xml#id(l_1321)"/>
</mw>
<mw id="m_003" type="V"> andato
  <lexit id="li_003" href="lex.xml#id(l_0053)"/>
</mw>
<mw id="m_004" type="PR"> a
  <lexit id="li_004" href="lex.xml#id(l_0001)"/>
</mw>
<mw id="m_005" type="N"> Roma
  <lexit id="li_005" href="lex.xml#id(l_5076)"/>
```

```
</mw>
<mw id="m_006" type="P"> io
  <lexit id="li_006" href="lex.xml#id(l_2385)"/>
</mw>
<mw id="m_007" type="PR"> a
  <lexit id="li_007" href="lex.xml#id(l_0001)"/>
</mw>
<mw id="m_008" type="N"> Parigi
  <lexit id="li_008" href="lex.xml#id(l_4503)"/>
</mw>
...
```

In this example, reference to lexical units in the file `lex.xml` is made possible through use of `<lexit>` elements, hierarchically embedded into `<mw>` elements.
`<lexit>` elements are specified for a unique identifier and a `href` pointer to the relevant lexical entry.

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001"                      href="mword.xml#id(mw_003)"/>
  <dep id="d_001" type="subj"           href="mword.xml#id(mw_001)"/>
  <dep id="d_002" type="iobj" intro="a" href="mword.xml#id(mw_008)"/>
</funct>
<funct id="funct_002">
  <head id="h_002"                      href="lex.xml#id(l_0053)"/>
  <dep id="d_003" type="subj"           href="mword.xml#id(mw_006)"/>
  <dep id="d_004" type="iobj" intro="a" href="mword.xml#id(mw_005)"/>
</funct>
...
```

### 5.5.2 Coordinating constructions

The core annotation outlined in section 5.3 above makes no provision for the treatment of coordinating constructions. In fact, given a sentence like *John and Mary went to the restaurant*, the subject relation between the verb *went* on the one side and the two nouns *John* and *Mary* on the other side is only expressed separately for each head:

subj(go,John)
subj(go,Mary)

In order to explicitly annotate that *John* and *Mary* form part of a coordinated construction we introduce here a set of special-purpose binary relations, namely AND, OR and COMMA, whose arguments are coordinated conjuncts.

*John and Mary went to the restaurant*

subj(go,John)
subj(go,Mary)
iobj(go,restaurant.<introducer="to">)
AND(John,Mary)

#### 5.5.2.1 Examples

(1) *John and Mary went to the restaurant*

mword.xml

```
...
<mw id="mw 001">John</mw>
```

```
<mw id="mw_002">and</mw>
<mw id="mw_003">Mary</mw>
<mw id="mw_004">went</mw>
<mw id="mw_005">to</mw>
<mw id="mw_006">the</mw>
<mw id="mw_007">restaurant</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
  <head id="h_001"                      href="mword.xml#id(mw_004)"/>
  <dep id="d_001" type="subj"           href="mword.xml#id(mw_001)"/>
  <dep id="d_002" type="subj"           href="mword.xml#id(mw_003)"/>
  <dep id="d_003" type="iobj" intro="to" href="mword.xml#id(mw_007)"/>
</funct>
<coord id="coord_001" type="and">
  <arg id="arg_001"                     href="mword.xml#id(mw_001)"/>
  <arg id="arg_002"                     href="mword.xml#id(mw_003)"/>
</coord>
...
```

(2) *Yesterday I talked to Martha, Philip and their grandmother*

mword.xml

```
...
<mw id="mw_001">yesterday</mw>
<mw id="mw_002">I</mw>
<mw id="mw_003">talked</mw>
<mw id="mw_004">to</mw>
<mw id="mw_005">Martha </mw>
<mw id="mw_006">,</mw>
<mw id="mw_007">Philip</mw>
<mw id="mw_008">and</mw>
<mw id="mw_009">their</mw>
<mw id="mw_010">grandmother</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
   <head id="h_001"                      href="mword.xml#id(mw_003)"/>
   <dep id="d_001" type="subj"           href="mword.xml#id(mw_002)"/>
   <dep id="d_002" type="mod"            href="mword.xml#id(mw_001)"/>
   <dep id="d_003" type="iobj" intro="to" href="mword.xml#id(mw_005)"/>
   <dep id="d_004" type="iobj"           href="mword.xml#id(mw_007)"/>
   <dep id="d_005" type="iobj"           href="mword.xml#id(mw_010)"/>
</funct>
<funct id="funct_002" >
   <head id="h_002"                      href="mword.xml#id(mw_010)"/>
   <dep id="d_006" type="mod"            href="mword.xml#id(mw_009)"/>
</funct>
<coord id="coord_001" type="comma">
   <arg id="arg_001"                     href="mword.xml#id(mw_005)"/>
   <arg id="arg_002"                     href="mword.xml#id(mw_007)"/>
</coord>
<coord id="coord_002" type="and">
   <arg id="arg_003"                     href="mword.xml#id(mw_007)"/>
   <arg id="arg_004"                     href="mword.xml#id(mw_010)"/>
</coord>
...
```

### 5.5.2.2 Markup

Markup of coordinating constructions is done by means of two dedicated elements, namely `<coord>` and `<arg>`. `<coord>` elements are very similar to `<funct>` elements, but they are further specified for an attribute `type`. Possible values for this attribute are `and`, `or` and `comma`. The element `<coord>` contains `<arg>` elements, which identify the lexical elements

that enter the coordinating relation. The markup tables for the <coord> and <arg> elements are illustrated below:

| <coord> | |
|---|---|
| id | [ASCII] |
| type | and, or, comma |

| <arg> | |
|---|---|
| id | [ASCII] |
| href | <mw> |

### 5.5.3 Binding constructions

In this section we deal with the annotation of two types of constructions, namely relative clauses and wh-clauses. In both cases, we will be concerned with extra-clause relations, as intra-clause relations have already been covered by the basic scheme.

For example, given a sentence like *the boy who I met yesterday is John*, in addition to the specification of all the usual dependency relations we may want to make explicit the relation between *boy* and the relative pronoun *who*. This represents an important piece of linguistic information, as it allows for an interpretation of *boy* as the intended direct object of the relative clause *who I met*.

In order to explicitly annotate the relation between *who* and *boy* we introduce here a binary relation called "bind", whose first slot is taken by the bound element (the relative pronoun in the case at hand), and the second slot by the binding noun:

bind(who,boy)

The same piece of annotation comes in handy for establishing an extra-clausal relationship between wh-interrogative pronouns in wh-questions and their respective replies in dialogue turns, as illustrated by the following example:

A: *what would you like to eat?*
B: *chocolate*

In an intra-clausal perspective, *what* is the object of *eat*. It would nonetheless be useful to keep record of the information that *chocolate* in turn B is understood as binding *what* in turn A, and is eventually a direct object of *eat*. In this case, we suggest to represent the relation between *what* and *chocolate* through a "bind" relation, as follows:

bind(what,chocolate)

A similar annotation can be applied to the following example:

A: *where did you go?*
B: *to Rome*

subj(go,you)
iobj(go,where)
bind(where,Rome.<introducer="to">)

### 5.5.3.1 Examples

(1) *the boy who I met yesterday is John*

mword.xml

```
...
<mw id="mw_001">the</mw>
<mw id="mw_002">boy</mw>
<mw id="mw_003">who</mw>
<mw id="mw_004">I</mw>
<mw id="mw_005">met</mw>
<mw id="mw_006">yesterday</mw>
<mw id="mw_007">is</mw>
<mw id="mw_008">John</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
   <head id="h_001"            href="mword.xml#id(mw_005)"/>
   <dep id="d_001" type="subj" href="mword.xml#id(mw_004)"/>
   <dep id="d_002" type="dobj" href="mword.xml#id(mw_003)"/>
   <dep id="d_003" type="mod"  href="mword.xml#id(mw_006)"/>
</funct>
<funct id="funct_002">
   <head id="h_002"            href="mword.xml#id(mw_007)"/>
   <dep id="d_004" type="subj" href="mword.xml#id(mw_002)"/>
   <dep id="d_005" type="comp" href="mword.xml#id(mw_008)"/>
</funct>
<bind id="bind_001">
   <arg id="arg_001"           href="mword.xml#id(mw_003)"/>
   <arg id="arg_002"           href="mword.xml#id(mw_002)"/>
</bind>
...
```

(2) *A: what would you like to eat?*
*B: chocolate*

mword.xml

```
...
<mw id="mw_001">what</mw>
<mw id="mw_002">would</mw>
<mw id="mw_003">you</mw>
<mw id="mw_004">like</mw>
<mw id="mw_005">to</mw>
<mw id="mw_006">eat</mw>
<mw id="mw_007">?</mw>
<mw id="mw_008">chocolate</mw>
...
```

funct.xml

```
...
<funct id="funct_001">
   <head id="h_001"                          href="mword.xml#id(mw_004)"/>
   <dep id="d_001" type="subj"               href="mword.xml#id(mw_003)"/>
   <dep id="d_002" type="comp" syn_real="x" href="mword.xml#id(mw_006)"/>
</funct>
<funct id="funct_002" >
   <head id="h_002"                          href="mword.xml#id(mw_006)"/>
   <dep id="d_003" type="dobj"               href="mword.xml#id(mw_001)"/>
</funct>
<bind id="bind_001">
   <arg id="arg_001"                         href="mword.xml#id(mw_001)"/>
   <arg id="arg_002"                         href="mword.xml#id(mw_008)"/>
</bind>
...
```

(3) *A: where did you go?*
*B: to Rome*

mword.xml

```
...
<mw id="mw_001">where</mw>
<mw id="mw_002">did</mw>
<mw id="mw_003">you</mw>
<mw id="mw_004">go</mw>
<mw id="mw_005">?</mw>
<mw id="mw_006">to</mw>
<mw id="mw_007">Rome</mw>
...
```

funct.xml

```
...
<funct id="funct_001" >
   <head id="h_001"            href="mword.xml#id(mw_004)"/>
   <dep id="d_001" type="subj" href="mword.xml#id(mw_003)"/>
   <dep id="d_002" type="iobj" href="mword.xml#id(mw_001)"/>
</funct>
<bind id="bind_001">
   <arg id="arg_001"           href="mword.xml#id(mw_001)"/>
   <arg id="arg_002"           href="mword.xml#id(mw_007)"/>
</bind>
...
```

#### 5.5.3.2 Markup

Markup of binding relations requires `<bind>` elements, which are very similar to `<funct>` and `<coord>` elements, except that they are not specified for any attribute apart from a unique identifier. `<bind>` elements contain `<arg>` elements, which identify the lexical elements that enter the binding relation. The markup table for the `<bind>` element is illustrated below. For `<arg>` elements, see section 5.5.2.1.

| <bind> | |
|---|---|
| id | [ASCII] |

**Annotation at the functional level**

Hereafter we provide an example of annotation at the functional level.

funct.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE funct_file SYSTEM "funct.dtd">
<funct_file>
<funct id="funct_001">
  <head id="h_001"                          href="mword.xml#id(mw_001)"/>
  <dep id="d_001"   type="dobj"             href="mword.xml#id(mw_003)"/>
  <dep id="d_002"   type="mod"              href="mword.xml#id(mw_005)"/>
</funct>
<funct id="funct_002">
  <head id="h_002"                          href="mword.xml#id(mw_005)"/>
  <dep id="d_003"   type="dobj"             href="mword.xml#id(mw_007)"/>
  <dep id="d_004"   type="iobj" intro="from" href="mword.xml#id(mw_010)"/>
  <dep id="d_004.5" type="iobj" intro="to"   href="mword.xml#id(mw_012)"/>
</funct>
```

```
<funct id="funct_003">
  <head id="h_003"                              href="mword.xml#id(mw_015)"/>
  <dep id="d_005"   type="subj"                 href="mword.xml#id(mw_017)"/>
  <dep id="d_006"   type="mod"                  href="mword.xml#id(mw_014)"/>
</funct>
<funct id="funct_004">
  <head id="h_004"                              href="mword.xml#id(mw_017)"/>
  <dep id="d_007"   type="mod"                  href="mword.xml#id(mw_018)"/>
</funct>
<funct id="funct_005">
  <head id="h_005"                              href="mword.xml#id(mw_021)"/>
  <dep id="d_008"   type="subj"                 href="mword.xml#id(mw_019)"/>
  <dep id="d_009"   type="dobj"                 href="mword.xml#id(mw_023)"/>
</funct>
<funct id="funct_006">
  <head id="h_006"                              href="mword.xml#id(mw_023)"/>
  <dep id="d_010"   type="mod"                  href="mword.xml#id(mw_022)"/>
  <dep id="d_011"   type="mod"                  href="mword.xml#id(mw_026)"/>
</funct>
<funct id="funct_007">
  <head id="h_007"                              href="mword.xml#id(mw_026)"/>
  <dep id="d_012"   type="mod"                  href="mword.xml#id(mw_027)"/>
</funct>
<funct id="funct_008">
  <head id="h_008"                              href="mword.xml#id(mw_029)"/>
  <dep id="d_013"   type="subj"                 href="mword.xml#id(mw_031)"/>
  <dep id="d_014"   type="mod"                  href="mword.xml#id(mw_028)"/>
</funct>
<funct id="funct_009">
  <head id="h_009"                              href="mword.xml#id(mw_033)"/>
  <dep id="d_015"   type="subj"                 href="mword.xml#id(mw_032)"/>
  <dep id="d_016"   type="comp"                 href="mword.xml#id(mw_035)"/>
  <dep id="d_017"   type="mod" intro="at"       href="mword.xml#id(mw_039)"/>
</funct>
<funct id="funct_010">
  <head id="h_010"                              href="mword.xml#id(mw_035)"/>
  <dep id="d_018"   type="mod"                  href="mword.xml#id(mw_034)"/>
  <dep id="d_019"   type="mod" intro="of"       href="mword.xml#id(mw_037)"/>
</funct>
<funct id="funct_011">
  <head id="h_011"                              href="mword.xml#id(mw_041)"/>
  <dep id="d_020"   type="subj"                 href="mword.xml#id(mw_044)"/>
  <dep id="d_021"   type="mod"                  href="mword.xml#id(mw_040)"/>
</funct>
<funct id="funct_012">
  <head id="h_012"                              href="mword.xml#id(mw_044)"/>
  <dep id="d_022"   type="mod"                  href="mword.xml#id(mw_042)"/>
</funct>
<funct id="funct_013">
  <head id="h_013"                              href="mword.xml#id(mw_046)"/>
  <dep id="d_023"   type="subj"                 href="mword.xml#id(mw_045)"/>
  <dep id="d_024"   type="comp"                 href="mword.xml#id(mw_048)"/>
</funct>
<funct id="funct_014">
  <head id="h_014"                              href="mword.xml#id(mw_048)"/>
  <dep id="d_025"   type="mod"                  href="mword.xml#id(mw_047)"/>
  <dep id="d_026"   type="mod"                  href="mword.xml#id(mw_050)"/>
</funct>
<funct id="funct_015">
  <head id="h_015"                              href="mword.xml#id(mw_046)"/>
  <dep id="d_027"   type="mod" intro="at"       href="mword.xml#id(mw_052)"/>
</funct>
<funct id="funct_016">
  <head id="h_016"                              href="lex.xml#id(l_00x)"/>
  <dep id="d_028"   type="subj"                 href="lex.xml#id(l_00x)"/>
  <dep id="d_029"   type="comp"                 href="mword.xml#id(mw_054)"/>
  <dep id="d_030"   type="mod"  intro="at"      href="mword.xml#id(mw_057)"/>
</funct>
</funct_file>
```

# References

Abney, S. 1996. *Chunk Stylebook*. Draft document. Available at http://www.sfs.nphil.uni-tuebingen.de/~abney/96i.ps.gz.

Burnage, G. 1990. *CELEX: A guide for users*. Technical Report, University of Nijmegen, Center for Lexical Information, Nijmegen.

Carroll, J., Briscoe, T., Calzolari, N., Federici, S., Montemagni, S., Pirrelli, V., Grefenstette, G., Sanfilippo, A., Carroll, G. and M. Rooth. 1997. *SPARKLE Work Package 1: Specifications of Phrasal Parsing*. Available at ftp://ftp.ilc.pi.cnr.it/pub/sparkle/deliv_1.ps.gz

Federici, S., S. Montemagni, V. Pirrelli, 1996, "Shallow Parsing and Text Chunking: a View on Underspecification in Syntax", in *Proceedings of the Workshop On Robust Parsing, European Summer School in Logic, Language and Information*. Prague, Czech Republic, 12-16 August 1996.

Federici, S., S. Montemagni, V. Pirrelli, 1998, "Chunking Italian: Linguistic and Task-oriented Evaluation". In J. Carroll (ed.), *Proceedings of the Workshop On the Evaluation of Parsing Systems*. Granada: Spain.

Leech, G., and A. Wilson. 1996. *EAGLES Recommendations for the Morphosyntactic Annotation of Corpora*. Available at http://www.ilc.pi.cnr.it/EAGLES96/annotate.annotate.html

Leech, G., M. Weisser, A. Wilson, 1998, *LE-EAGLES WP4, 3.1, Integrated Resources Working Group, Draft Chapter: Survey and Guidelines for the Representation and Annotation of Dialogue*.

Levelt, W.J.M. 1989. *Speaking: from Intention to Articulation*. Cambridge, MA: MIT Press.

MacWhinney, B., 1991, *The CHILDES project: Tools for analyzing talk*. Hillsdale, NJ: Erlbaum.

MacWhinney, B., 1994, *The CHILDES Project: Tools for Analyzing Talk*, 2nd edition. Hillsdale, NJ: Erlbaum.

Meeter, M. et al. 1995. *Dysfluency Annotation Stylebook for the Switchboard Corpus*. February 1995. Available at ftp://ftp.cis.upenn.edu/pub/treebank/swbd/doc/DFL-book.ps

Sampson, G, 1995, *English for the Computer*, Oxford: Clarendon Press.

The CHRISTINE Corpus. http://www.susx.ac.uk/users/geoffs/Rchristine.html

# Dialogue Act Markup

Marion Klein (DFKI)

## 1.Introduction

The scope of this chapter is the level of dialogue acts and its markup. Dialogue acts are also referred to as moves, or illocutionary acts. They mark important characteristics of utterances, indicate the role or intention of an utterance in a specific dialogue, and make relationships between utterances more obvious.
Base units of dialogue acts are utterances or segments. They are considered as sequences of words. Utterances/segments are sub components of turns which are speaking units of dialogue partners.
Dialogue act annotation is used for training and testing purposes of NLP systems. One can derive statistical predictions about follow-up dialogue acts based on previous dialogue act annotations and on pattern recognition to improve performance of recognisers. Furthermore, dialogue act language models serve to study intonation, and to analyse referring expressions.

In the following we summarize the results of scheme comparison in D1.1 to reflect the state of the art on this level. Based on this information we suggest best practice methods for scheme design which fulfil the requirements derived from the comparison. Finally, an example-markup and further supported schemes are stated.

## 2. Existing Schemes

In D1.1 we compared 16 different schemes, developed in the UK, Sweden, the US, Japan, the Netherlands and Germany. Not all of these schemes can be annotated reliably and are suitable for reuse. Therefore some selecting criteria are needed to find out which schemes are appropriate:

***Criteria*** (short version of D1.1)***:***

1. Existence of coding book.
2. Number of annotators.
3. Level of annotator's expertise.
4. Number of dialogues annotated.
5. Evaluation of scheme.
6. Underlying task.
7. Languages applied to.
8. Used in NLP systems.

The table below gives a brief overview of the comparison. More details can be found in D1.2.

| schemes | criteria | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Alparon | + | 3 | experts | 500 d. | 77% agreem. | DES | D | + |
| Chat | + | huge | experts | 160 MB | - | - | many | - |
| Chiba | + | 10 | experts | 22 d. | 0.57<a<0.68 | DIR,BA,TR | J | ? |
| Coconut | + | 2 | experts | 16 d. | + | FUR | E | + |
| Condon & Cech's | + | 5 | fairly exp. | 88 d. | 91% agreem. | TS | E | + |
| C-Star | + | 5 | experts | 230 d. | - | TR | E, J, K, I | + |
| DAMSL | + | 4 | experts | 18 d. | k = 0.56 | - | E | + |
| Flammia's | + | 7 | trained | 25 d. | k >= 0.6 | DES | E | ? |
| Janus | + | 4 | experts | many | 89% agreem. | BA | E | + |
| Linlin | + | 4 | experts | 140 d. | 97% agreem. | TR,TS | S | + |
| Map Task | + | 6 | experts | 128 d. | k = 0.83 | DIR | E | + |
| Nakatani's | + | 6 | naive | 72 d. | - | INSTR | E | + |
| SLSA | + | 7 | experts | 100 d. | + | COU | S | + |
| SWBD-DAMSL | + | 9 | experts | 1155 d. | 0.8<k<0.84 | - | E | + |
| Traum's | + | 3 | experts | 36 d. | + | - | E | + |
| Verbmobil | + | 3 | naive | 1172 d. | k = 0.84 | BA | E, J, G | + |

Overview of Scheme Comparison

**Abbreviations:**

| | |
|---|---|
| BA | business appointment |
| COU | courtroom interaction |
| DES | directory enquiry service |
| DIR | giving direction |
| FUR | furnishing rooms interaction |
| INSTR | giving instruction (e.g. about cooking) |
| TS | transport |
| TR | travel |
| | |
| D | Dutch |
| E | English |
| G | German |
| I | Italian |
| J | Japanese |
| K | Korean |
| S | Swedish |

The results of the comparison are:

- Coding books are provided for all schemes so that they can be used by other coders.
- All schemes seem to be applicable as they were applied to corpora of reasonable size by several annotators. Additionally, most of the schemes are used in NLP systems.
- Most schemes seem to be difficult to use as annotators were trained or experts.

- Schemes which state inter-coder agreement show intermediate to good results and are reliable.
- Most of the schemes are hard to reuse in a different context because they are domain/task or language dependent.

The last item of the list is crucial. At the moment it is not possible to come up with a general list of phenomena for the level of dialogue acts which would suit everybody's requirements. What phenomena a scheme models depends on the research interests of a scheme developer which vary to a great extend.

# 3. Best Practice for Dialogue Act Scheme Design

Instead of proposing *the scheme* for the level of dialogue acts we would rather like to point scheme developers to a best practice method as considered by the Discourse Resource Initiative (DRI):

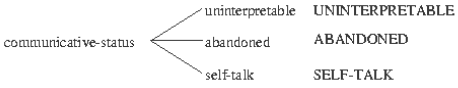
1. Make a list of all phenomena you are interested in and assign a certain characteristic set of tags to each phenomena.
2. Use this multi-dimensional scheme by yourself, apply it to some training corpora and test reliability.
3. Flatten the multi-dimensional scheme to a single-dimensional one, by merging tags which always occur together and deleting those which were never used. Remember not to have any extremely small categories as coders tend to overuse them and use natural, easy observable distinctions of tags which are easy to remember.
4. Provide a coding book for the single-dimensional scheme, including tag set definition, decision tree and example annotations.
5. Develop a mapping mechanism to convert multi-dimensional annotation to single-dimensional annotation and the other way around.
6. Randomly check the coding by reliability tests and improve your scheme(s), if necessary.
7. Document all steps!

The advantage of this method is that the multi-dimensional scheme supports reusability because it models the different phenomena best, and thus, is easier to understand by foreign scheme developers. The single-dimensional scheme, on the other hand, is easier to apply which speeds-up the annotation process and hence, makes annotation less expensive. For these reasons the flattened scheme is much more appropriate for mass data annotation.
Of course, a flattened scheme is not necessarily required and if coders are happy with multi-dimensional coding, the time-consuming process of flattening (3.) can be omitted.

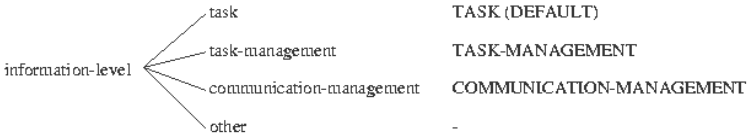
# 4. Markup

The approach in MATE is to reuse the DAMSL scheme as an example for a multi-dimensional scheme and a variant of SWBD-DAMSL as its example flattened counterpart. SWBD-DAMSL was derived from the original DAMSL scheme using the techniques described above. Unfortunately some additional tags were added so that an exact mapping from one scheme to the other is not possible any more. For this reason the MATE SWBD-DAMSL variant omits these additional tags. The following graphics shows the relation between the tag sets, where DAMSL tags are typed in small letters and SWBD-DAMSL tags are typed in capital letters:

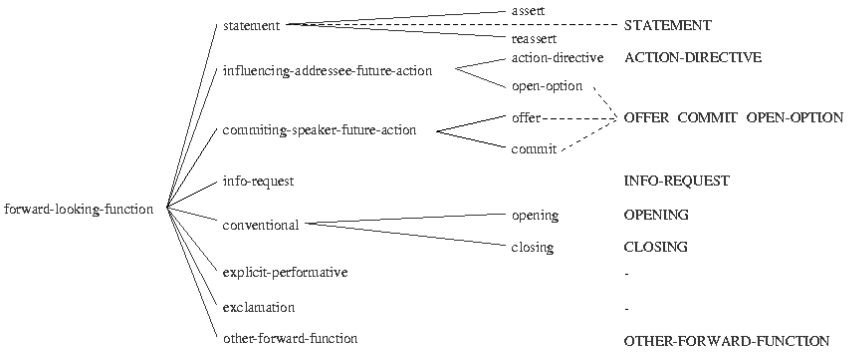
- communicative-status



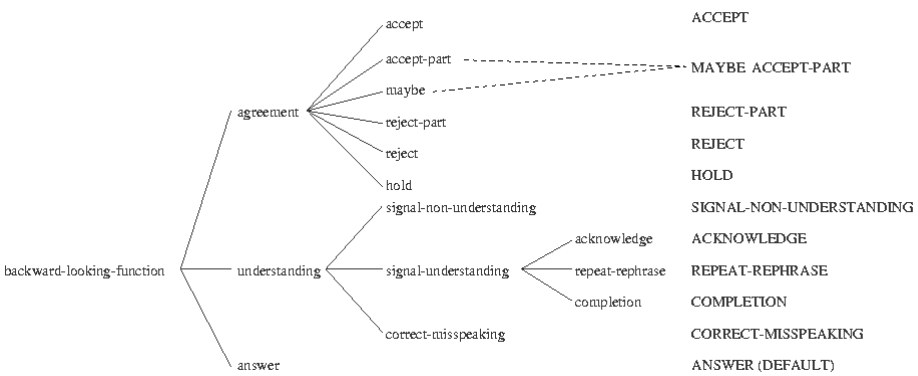

- information-level




- forward-looking-function




- backward-looking-function



For information on the tag set description we want to refer to the coding books of DAMSL and SWBD-DAMSL.
All dialogue act tags are XML elements and are children of a basic dialogue act unit, called segment. Such a unit is the amount of material that can be attributed one dialogue act / illocutionary function. Each element has as its attributes an unique identifier (id) and segment elements have in addition a reference to the word level (href), the base level for dialogue acts.

The mapping mechanism from the internal DAMSL scheme to its surface counterpart, a variant of the SWBD-DAMSL scheme, is realised using MATE style sheets. Mapping can be performed in both directions. The relevant stylesheets are presented in the appendix (internal structure to surface structure stylesheet, surface structure to internal structure stylesheet).

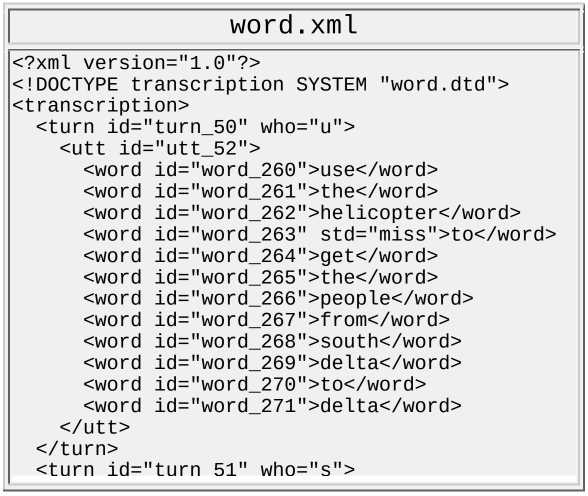
# 5. Other Supported Schemes

Other schemes which will be supported in the MATE workbench are the Map Task and the Verbmobil scheme. Both schemes are well-documented and successfully reliability tested. They belong to the most well-known schemes and serve in the MATE workbench as successfully used, task-oriented and domain-dependent example schemes. Details about the tagsets of both schemes can be found in their coding-books (Map Task coding book, Verbmobil coding book) and in their coding modules (Map Task coding module, Verbmobil coding module).

# 6. Summary

Based on the great variety of schemes we compared in D1.1 we presented a method of coding scheme design that supports reusability of schemes and also leads to reliable mass data annotation. We used the DAMSL and a variant of the SWBD-DAMSL scheme to exemplify our proposal. DTDs, example annotations, and conversion stylesheets in the appendix supplement our approach. As additional examples, the Map Task and Verbmobil schemes are implemented in the workbench.

### Dialogue Acts: Annotation example



word.xml

```
<?xml version="1.0"?>
<!DOCTYPE transcription SYSTEM "word.dtd">
<transcription>
  <turn id="turn_50" who="u">
    <utt id="utt_52">
      <word id="word_260">use</word>
      <word id="word_261">the</word>
      <word id="word_262">helicopter</word>
      <word id="word_263" std="miss">to</word>
      <word id="word_264">get</word>
      <word id="word_265">the</word>
      <word id="word_266">people</word>
      <word id="word_267">from</word>
      <word id="word_268">south</word>
      <word id="word_269">delta</word>
      <word id="word_270">to</word>
      <word id="word_271">delta</word>
    </utt>
  </turn>
  <turn id="turn 51" who="s">
```

```
    <utt id="utt_53">
      <word id="word_272">alright</word>
    </utt>
  </turn>
  <turn id="turn_52" who="u">
    <utt id="utt_54">
      <word id="word_273">where</word>
      <word id="word_274">are</word>
      <word id="word_275">the</word>
      <word id="word_276">people</word>
      <word id="word_277" std="ins">left</word>
    </utt>
  </turn>
  <turn id="turn_53" who="s">
    <utt id="utt_55">
      <word id="word_278">i</word>
      <word id="word_279">don't</word>
      <word id="word_280">understand</word>
      <word id="word_281">your</word>
      <word id="word_282">reference</word>
      <word id="word_283">to</word>
      <word id="word_284">the</word>
      <word id="word_285">people</word>
      <word id="word_286">left</word>
    </utt>
  </turn>
  <turn id="turn_54" who="u">
    <utt id="utt_56">
      <word id="word_287">where</word>
      <word id="word_288">are</word>
      <word id="word_289">the</word>
      <word id="word_290">people</word>
    </utt>
  </turn>
  <turn id="turn_55" who="s">
    <utt id="utt_57">
      <word id="word_291">there</word>
      <word id="word_292">are</word>
      <word id="word_293">2</word>
      <word id="word_294">groups</word>
      <word id="word_295">of</word>
      <word id="word_296">people</word>
      <word id="word_297">at</word>
      <word id="word_298">delta</word>
    </utt>
  </turn>
  <turn id="turn_56" who="u">
    <utt id="utt_58">
      <word id="word_299">where</word>
      <word id="word_300">are</word>
      <word id="word_301">all</word>
      <word id="word_302">the</word>
      <word id="word_303">people</word>
    </utt>
  </turn>
  <turn id="turn_57" who="s">
    <utt id="utt_59">
      <word id="word_304">there</word>
      <word id="word_305">are</word>
      <word id="word_306">6</word>
      <word id="word_307">groups</word>
      <word id="word_308">of</word>
      <word id="word_309">people</word>
      <word id="word_310">at</word>
      <word id="word_311">delta</word>
      <word id="word_312">and</word>
      <word id="word_313">1</word>
      <word id="word_314">at</word>
      <word id="word_315">south</word>
      <word id="word_316">delta</word>
    </utt>
  </turn>
```

```
         <task id="x2"/>
      </information_level>
      <forward_looking_function id="flf_action_directive__x3">
         <influencing_addressee_future_action id="iafa_action_directive__x3">
            <action_directive id="x3"/>
         </influencing_addressee_future_action>
      </forward_looking_function>
   </segment>
   <segment id="x4" href="word.xml#id(word_272)">
      <information_level id="il_communication_management__x5">
         <communication_management id="x5"/>
      </information_level>
      <backward_looking_function id="blf_acknowledge__x6">
         <understanding id="u_acknowledge__x6">
            <signal_understanding id="su_acknowledge__x6">
               <acknowledge id="x6"/>
            </signal_understanding>
         </understanding>
      </backward_looking_function>
   </segment>
   <segment id="x7" href="word.xml#id(word_273)..id(word_277)">
      <information_level id="il_task__x8">
         <task id="x8"/>
      </information_level>
      <forward_looking_function id="flf_info_request__x9">
         <info_request id="x9"/>
      </forward_looking_function>
   </segment>
   <segment id="x10" href="word.xml#id(word_278)..id(word_286)">
      <information_level id="il_communication_management__x11">
         <communication_management id="x11"/>
      </information_level>
      <forward_looking_function id="flf_statement__x12">
         <statement id="x12"/>
      </forward_looking_function>
      <forward_looking_function id="flf_offer__open_option__x13">
         <committing_speaker_future_action id="csfa_offer__open_option__x13">
            <offer id="offer__open_option__x13"/>
         </committing_speaker_future_action>
      </forward_looking_function>
      <forward_looking_function id="flf_commit__open_option__x13">
         <committing_speaker_future_action id="csfa_commit__open_option__x13">
            <commit id="commit__open_option__x13"/>
         </committing_speaker_future_action>
      </forward_looking_function>
      <forward_looking_function id="flf_open_option__open_option__x13">
         <influencing_addressee_future_action id="iafa_open_option__open_option__x13">
            <open_option id="open_option__open_option__x13"/>
         </influencing_addressee_future_action>
      </forward_looking_function>
      <backward_looking_function id="blf_signal_non_understanding__x14">
         <understanding id="u_signal_non_understanding__x14">
            <signal_non_understanding id="x14"/>
         </understanding>
      </backward_looking_function>
   </segment>
   <segment id="x15" href="word.xml#id(word_287)..id(word_290)">
      <information_level id="il_task__x16">
         <task id="x16"/>
      </information_level>
      <forward_looking_function id="flf_info_request__x17">
         <info_request id="x17"/>
      </forward_looking_function>
      <backward_looking_function id="blf_answer__x18">
         <answer id="x18"/>
      </backward_looking_function>
   </segment>
   <segment id="x19" href="word.xml#id(word_291)..id(word_298)">
      <information_level id="il_task__x20">
         <task id="x20"/>
      </information_level>
      <forward_looking_function id="flf_statement__x21">
         <statement id="x21"/>
      </forward_looking_function>
      <backward_looking_function id="blf_answer__x22">
         <answer id="x22"/>
      </backward_looking_function>
   </segment>
   <segment id="x23" href="word.xml#id(word_299)..id(word_303)">
      <information_level id="il_task__x24">
         <task id="x24"/>
      </information_level>
      <forward_looking_function id="flf_info_request__x25">
         <info_request id="x25"/>
      </forward_looking_function>
   </segment>
   <segment id="x26" href="word.xml#id(word_304)..id(word_316)">
      <information_level id="il_task__x27">
```

```
         <task id="x27"/>
      </information_level>
      <forward_looking_function id="flf_statement__x28">
         <statement id="x28"/>
      </forward_looking_function>
      <backward_looking_function id="blf_answer__x29">
         <answer id="x29"/>
      </backward_looking_function>
   </segment>
</da_internal_file>
```

## Dialogue Acts: Example Annotation for the surface scheme

da_surfce.xml

```
<?xml version='1.0'?>
<!DOCTYPE da_surface_file SYSTEM "da_surface.dtd" []>
<da_surface_file>
   <segment id="x1" href="word.xml#id(word_260)..id(word_271)">
      <task id="x2"/>
      <action_directive id="x3"/>
   </segment>
   <segment id="x4" href="word.xml#id(word_272)">
      <communication_management id="x5"/>
      <acknowledge id="x6"/>
   </segment>
   <segment id="x7" href="word.xml#id(word_273)..id(word_277)">
      <task id="x8"/>
      <info_request id="x9"/>
   </segment>
   <segment id="x10" href="word.xml#id(word_278)..id(word_286)">
      <communication_management id="x11"/>
      <statement id="x12"/>
      <offer___commit___open_option id="open_option__x13"/>
      <signal_non_understanding id="x14"/>
   </segment>
   <segment id="x15" href="word.xml#id(word_287)..id(word_290)">
      <task id="x16"/>
      <info_request id="x17"/>
      <answer id="x18"/>
   </segment>
   <segment id="x19" href="word.xml#id(word_291)..id(word_298)">
      <task id="x20"/>
      <statement id="x21"/>
      <answer id="x22"/>
   </segment>
   <segment id="x23" href="word.xml#id(word_299)..id(word_303)">
      <task id="x24"/>
      <info_request id="x25"/>
   </segment>
   <segment id="x26" href="word.xml#id(word_304)..id(word_316)">
      <task id="x27"/>
      <statement id="x28"/>
      <answer id="x29"/>
   </segment>
</da_surface_file>
```

# COREFERENCE

*Massimo Poesio*

# 1 Coding Purpose

In this chapter we present the coding schemes for coreference in dialogues supported in the MATE project.

## 1.1 What is 'coreference'?

The term 'coreference annotation' is used in an informal way in corpus work to indicate both the annotation of (generalized) anaphoric information and of information about reference proper. We use the term **Anaphoric Relation** to indicate the relation between two textual elements that denote the same object; the subsequent mention of an entity already introduced is often marked by means of a particular type of noun phrase (NP) called an **anaphoric expressions.** Annotating corpora with information about such relations between elements of a text is useful both from a linguistic point of view and for applications such as information extraction. A typical example of anaphoric expression are pronouns such as *he* in the text

*John arrived. He looked tired.*

In the preferred reading of this text, the pronoun *he* is a sort of 'abbreviated mention' of the individual 'John' which is denoted by the expression *John*. Following the terminology introduced by Sidner (1979) we will say that in the example just discussed the pronoun *he* **co-specifies** with the proper name *John*, and we will call John the **antecedent** of the pronoun. We will also say that two strings **co-refer** when they point to the same entity in the world. In the example above, the pronoun *he* and the proper name *John* both co-specify and co-refer; more in general, two expressions may co-specify without co-referring, as we will see below.

The notion of anaphora just introduced is often generalized to relations other than identity. So-called **bridging references** (Clark, 1977) are expressions that denote objects only related to the denotation of their antecedent by (shared) generic knowledge. An example is *the indicators* in:

*John has bought a new car. The indicators use the latest laser technology.*

We are able to interpret the description *the indicators* because we know that indicators are a part of cars, and a car was mentioned in the first sentence. Some of the relations that may hold between a bridging reference and its antecedent include part-whole as in the example just seen, and element-set (as in *The Italian team didn't play well yesterday until the centre-forward was replaced in the 30th minute*). A bridging reference may also refer to the object filling a role in an event, whether implicitly or explicitly introduced, e.g. *A young woman was attacked earlier this evening on Town Moor. The assailant was chased by a member of the public, but managed to escape.* (A detailed survey of alternative classifications of bridging descriptions proposed in the literature can be found in Vieira (1998).)

Another example of expression which has an 'antecedent', but whose relation with the antecedent is not of identity, is the expression *one* in *Wendy prefers the red T-shirt to the yellow one*. In this case, we are talking about two distinct T-shirts, of different colours. The expression *one* thus denotes something like an object type rather than an object token. Pronouns can enter in the same

type of semantic relation with their antecedents, albeit more rarely: the classical example of this are sentences such as *The man who gave his paycheque to his wife was wiser than the man who gave it to his mistress*, which give this kind of pronouns the name **paycheck pronouns**. Yet another example of indirect relation between an anaphoric expression and its antecedent are **bound** pronouns (Partee, 1972). In *Nobody likes to lose his job*, the pronoun *his* does not 'refer' to the same object as its antecedent, the quantifier *nobody* (which does not refer to anything); this anaphoric expression is best seen as playing the role of a variable in first order logic.

So far, we have seen examples of anaphoric expressions which refer back to an object introduced in the text, or are somehow related to it (as in the case of bridging references). However, for some applications (especially multimedia ones) it is also useful to mark the cases in which an expression in the text refers to an object that has not been mentioned before, but is 'accessible' because it is part of the visible situation: these expressions are called **deictics** or also **indexicals**. An example of indexical expression in a real life conversation is *the salt* in an utterance of the sentence *pass me the salt, please* in a context in which the salt hasn't been mentioned before. The MapTask corpus collected at HCRC contains a number of references to so-called 'landmarks' - objects on a map that the participants in a conversation look at while doing the task - which are also deictic in this sense, as are the references to objects on the screen in the GOCAD corpus from LORIA.

## 1.2 Issues to be considered in a dialogue coreference annotation scheme

Whether one is working on text or dialogue, the main problem in annotating anaphora is that almost every word in a text may be anaphoric (in the generalized sense discussed above) to some extent; hand-annotating all anaphoric expressions and all anaphoric relations is therefore impossible, except for small amounts of text. When designing a scheme for annotating anaphoric relations it is then necessary to identify the anaphoric expressions and relations more relevant for one's needs. Narrowing the scope of the scheme may also be necessary in order to achieve good agreement among subjects.

This can be done by specifying syntactic constraints on markables, which are the text spans which enter into coreference relationships, by specifying constraints on the sorts of objects in the world for which coreference will be marked up, or by restricting the kinds of coreferential relations which will be considered (for instance, by deliberately failing to mark bridging references). In addition to the problem of what counts as a markable, there are additional difficulties which are thrown up by annotating dialogue instead of text: what to do about marking up coreferences which occur during disfluent speech, and what to do if the participants in a dialogue do not agree about what an expression refers to, especially if they know about different objects in the world.

### 1.2.1 Syntactic restrictions on markables

One way of limiting the annotation task is to use syntactic restrictions to determine a set of text spans which the coder will then consider as markables for coreference relations. For instance, many schemes restrict mark-up to NPs, whether these are determined by the human coder or automatically via a morphosyntactic tagger. And even so, the choice of NPs to serve as markables is not straightforward. For instance, it is quite common to ignore first and second person pronouns when marking. It is not clear whether to mark appositions in noun phrases separately (as in "*one*

*of engines at Elmira, say engine E2* " or "*The Admiral's Head, that famous Portsmouth hostelry* "). Similarly, noun phrases in post-copular position can be problematic. For example, it can be argued that in (1.1) *a policeman* is clearly expressing a predicate, and therefore need not be marked, whereas in (1.2) (to be imagined being said while looking at the sky at night), both *the planet on the left* and *Venus* are clearly referring expressions; it's not so clear how to handle *the president of the board* in (1.3).

(1.1) *John is a policeman*.

(1.2) *The planet on the left is Venus*.

(1.3) *John is the president of the board*.

It may be useful to mark empty elements such as that seen in *Sieve the flour and baking powder into the fat. Mix _.*, even though they leave no trace in the words of the transcript. Anaphoric references to events and other abstract objects may also stretch the notion that markables are traceable NPs.

An issue that has to be considered when thinking about other languages is that in languages such as Spanish and Italian, anaphoric expressions may be morphologically incorporated in the verb: In Italian, for example, certain clitics behave like verb suffixes:

(1.4) *A: Adesso dammelo.* [*Now give-to me-it]*

Because the most common syntactic constructions for coreferential expressions differ in different languages, because people may wish to use different syntactic constraints for different purposes, and because, even with the same purposes, people use different automatic morphosyntactic taggers which make different syntactic distinctions, it is not sensible to impose any standard views on the correct syntactic constraints to use for pre-filtering possible markables. As a result, our approach is to allow the user of the MATE workbench to decide upon a syntactic constraint which suits their corpus and their automatic tagging, by expressing it in the MATE query language. Users who do not wish to impose syntactic constraints at all (for instance, those interested in determining what the distribution of syntactic constructions are for the different kinds of coreference relations) may specify a null constraint, in which case the human coder must scan the complete text looking for referring expressions to code.

### 1.2.2 Choosing an object type constraint on markables

As well as using syntactic constraints to cut down on the number of coreference annotations, it is also possible to specify restrictions on the kinds of objects in the world for which coreference is of interest. For instance, in the Map Task, researchers often want to know about coreference relations for map landmarks but not for anything else. As with syntactic constraints, reasonable object type constraints will depend on the material being marked. Therefore, again our approach is to allow the user of the MATE workbench to specify this constraint, either as a pre-determined list of objects or by giving a description of the objects of interest. In this latter case, it is of course impossible for the workbench itself to determine which text spans fit the constraint, and so this constraints forms part of the coding instructions for the human user to follow.

### 1.2.3 Restricting the coreference relations to be marked

Another way of limiting the coreference annotation task is to ask the coder only to mark some kinds of coreference relations. For instance, the very simplest coreference schemes, like MUCCS (Hirschman, 1997) and the scheme used in the Map Task, only specify a relationship when the two discourse entities being linked refer to the same object. One good reason for limiting coreference annotation exercises by restricting the set of relations to be marked is that for many of the most interesting relations, reliable annotation schemes have not yet been developed. The best reliability information to date comes from work by Poesio and Vieira (1997), which concentrated on marking definite descriptions on texts from the Wall Street Journal. Their results confirm Fraurud's (1990) impression that the only distinction that can be marked reliably is that between first mentions and subsequent mentions; bridging references proved remarkably difficult to classify reliably. Of course, for many purposes, and especially for linguistic research on the role of bridging, even unreliable coding may be valuable; however, for large-scale annotation exercises with a language engineering bent, a simpler set of relations may be more appropriate.

### 1.2.4 Deciding what to do about disfluencies

When annotating dialogues, new problems arise, one of which is what to do about hesitations and disfluencies (such as repetitions and repairs), which break up the syntax of an utterance and can occur in the same location as a referring expression. In (1.5) (from the TRAINS corpus, (Gross et al, 1993)), the noun phrase *one of engines at Elmira, say engine E2* is divided between several utterances, broken by pauses and other hesitations. In (1.6) (from (Passonneau, 1996)), the definite description *the other kids* is repaired into *the kid*.

(1.5)

9.6: I think what we should do

9.7: is

9.8: hook up

9.9: uh one of the [2sec]

9.10: engines

9.11: uh

9.12: at Elmira

9.13: say engine E2

(1.6) *and the g guy on the bike gives the other kids... gives the kid that returns his hat...*

This can cause difficulties for syntactic constraints on markables unless the morphosyntactic tagging takes disfluency into account by splicing disfluent utterances into their perceived targets. What one chooses to do about disfluency is likely to depend on the expected use of the coreference tagging and what possibilities the morphosyntactic tags leave open. If the morphosyntactic tagging allows one to splice together target utterances, then one might choose to ignore disfluencies by constructing and marking on these targets. Alternatively, one might choose to ignore all possible markables within disfluent speech.

### 1.2.5 Multiple perspectives and misunderstandings

Another problem with annotating coreference in dialogues is that the participants do not always share the same perspective of the world or of the discourse. Sometimes different participants know about different objects in the world, leading to difficulties when one refers to an object unknown to the other. The Map Task makes this obvious by establishing differences between the participants' maps, but some knowledge differences occur in most real-world situations. Even where the universe of objects is completely shared, misunderstandings can arise because people are not always very careful in establishing joint references. As a result, different participants may believe that different coreference relations hold for the same markables. It is possible to allow the annotation of multiple perspectives within a dialogue, if one both allows multiple universes of objects, so that differences in world knowledge are clear, and allows the marking of coreferential links with the set of participants for which they hold. However, this does make annotation rather more complicated than it would be otherwise, and the annotation itself may not be particularly reliable, since making these distinctions requires a certain amount of mind-reading on the part of the coder. Another possibility is to specify that the coder is to annotate only the interpretation of a given noun phrase intended by the speaker. This still requires mind-reading, but less, since only one participant's mind must be read and since the speaker leaves the largest trace of what they think in the transcript.

## 1.3 Sources of Examples

A few examples in this document are made up, but most of them come from three main corpora:

- The MapTask corpus of direction-giving dialogues, collected at the University of Edinburgh (Anderson et al, 1991);
- The Microfusées corpus of French dialogues in a multi-media domain, collected at LORIA by Florence Bruneseaux and Laurent Romary;
- The TRAINS corpus of task-oriented dialogues, collected at the University of Rochester (Gross et al, 1993).

In addition, we took several examples from (Quirk and Greenbaum, 1973), from Passonneau's manual (Passonneau, 1996) and from the BBC News web site. We indicate the source of the examples either by explicitly mentioning the source or by means of the symbols (BBC) for the BBC texts, (MF) for the Microfusées texts, (QG) for Quirk and Greenbaum, and (T) for the TRAINS texts.

# 2 Existing Schemes

We analyzed five existing schemes in preparation for this proposal. Although in general MATE chose to review only schemes which had been proven reliable, in the case of coreference, reliability tests were rare or informal enough that this constraint was somewhat relaxed. The five schemes reviewed were the MUCSS scheme developed for MUC-7 (Hirschman, 1997), the DRAMA scheme (Passonneau, 1996), the Lancaster University UCREL scheme (Fligelstone, 1992), the scheme developed by Bruneseaux and Romary (1997) and the MapTask annotation of landmarks. These schemes are discussed in MATE deliverable D1.1.

The MUCCS scheme is the best known and most widely used of the existing coreference schemes, the more modest in scope (it concentrates on identity relations between NPs) and the only one whose reliability has been systematically tested. However, this scheme was designed for text, so it does not provide instructions either for dealing with problems in dialogue such as disfluencies or misunderstandings, or for annotating references to the visual situation, common e.g., in the MapTask corpus and in multimodal applications, and that we hypothesize can be reliably annotated. Also, its syntactic constraint on markables is designed only for English. The DRAMA scheme was designed for dialogues and therefore does include instructions for dealing with some difficult problems of markable identification in dialogues, but still relies on English-specific syntactic constraints in order to reduce the annotation task to something doable. DRAMA also includes instructions for dealing with bridging references - whose reliability however still has to be ascertained - but not for references to the visual situation. Finally, the Lancaster scheme was also designed for texts, and in certain ways is more ambitious than any of the schemes discussed here in that it also contains instructions for annotating elliptical references. We are not aware of any study of the reliability of the scheme.

# 3 Selected Schemes

Given that most coreference work is currently done with schemes which are not particularly reliable, and that there is little general agreement on the names of relations to use, we have adopted a modular approach to coreference schemes by which users can construct a scheme which is appropriate for them. Because the semantics of anaphora and coreference is relatively well-understood, it is possible to extract from the schemes discussed above a fairly short list of options available to the designer of a scheme. (This is unlike the case of dialogue acts, where different schemes are very difficult to compare.) These considerations suggested a 'meta-scheme' approach to the problem of developing a scheme for the coreference level that could be useful for a variety of applications. What this means is that instead of proposing a single scheme, we identified a range of types of information about 'coreference' that the designer of a scheme may want to annotate among those specified in the coding schemes for coreference discussed above; we evaluated how reliable each type of annotation is likely to be; and we specified the markup language needed to pursue each option. The workbench will support the whole range of elements and attributes of the meta-scheme; the task of the designer of a scheme will be to identify the options of interest among those supported by the workbench, ignoring the rest. Specifying a specific coreference scheme out of this range of options involves specifying syntactic and object type constraints, both of which can be empty, which are to be used for pre-filtering markables, plus specifying the coreference relations of interest, which will be assumed to be defined meaningfully for the human user and documented within the coding module.

In order to show that this approach is workable, in addition to providing a specification of this range of options we also showed how a useful set of schemes can be specified using the elements and attributes of the meta-scheme, so that the coreference community can use the MATE workbench to annotate according to their favorite scheme. These schemes include a 'basic' scheme that can be used to do the type of annotation that is done using MUCCS; a scheme that can be used to annotate references to the visual situation, as in the MapTask scheme and in the scheme developed by Bruneseaux and Romary; and a scheme to do the type of annotation which is possible in DRAMA, which involves an extended set of anaphoric relations. In addition, we included a discussion of the possible options when selecting markables, including instructions for annotating anaphoric constructs typical in Romance languages such as clitics, and for dealing with some typical dialogue phenomena. (This discussion does not cover all possible sorts of anaphoric

relations and uses of deictics; it is only concerned with the cases in which the anaphoric expression is an NP.)

# 4 Markup Elements common to all Options

In this section we introduce the common core of markup distinctions common to all options allowed by the meta-scheme.

## 4.1 Markup Declaration

The following elements are used in the coreference schemes. As in all other schemes we use a single element to mark both anaphoric expressions and the NPs that serve as antecedents; the main difference from the MUC-7 scheme and DRAMA is that, following Bruneseaux and Romary (1997) (who, in turn, followed the TEI specification), we separated out the annotation of co-specification from the annotation of discourse entities. We use therefore two main elements: `<coref:de>`, used to annotate the elements which enter in co-specification relations; and `<coref:link>`, used for expressing co-specification between discourse entities. This way of annotating relations has the advantage that a discourse entity can be related by links to more than one other discourse entity; this is important to allow a discourse entity to be related both to an antecedent introduced in the discourse and to an entity in the universe of discourse. In addition, we have elements for specifying objects in the visual situation that can serve as antecedents, and for marking text constituents that introduce elements which participate in anaphoric relations in an indirect way.

- `<coref:de>`: this tag is used to mark every text span that may enter in an anaphoric relation.

  Attributes:

  - ID
  - HREF

- `<coref:link>`: this tag is used to indicate anaphoric relations.

  Attributes:

  - HREF (obligatory)
  - TYPE (obligatory; with values as specified under the schemes)
  - WHO-BELIEVES (optional; default value SHARED; other values to be set to the participants in the dialogue (below, G and F)).

  Embedded elements: `<coref:anchor>`

- `<coref:anchor>`: this tag is used to indicate the antecedent in an anaphoric relation.

  Attributes:

  - HREF

- <coref:universe>: this element is used to introduce the objects in the visual situation of discourse.

  Attributes:

  - ID (obligatory)
  - modifies (optional, only permitted value is COMMON, used when the universe extends the common universe

  Embedded elements: <coref:ue>
- <coref:ue>: there should be one such element for each object in the universe.

  Attributes:

  - ID

  <coref:seg>: this tag is used to mark elements that participate in anaphoric relations but are not expressed by NPs, such as incorporated clitics in Romance languages and the antecedents of discourse deixis.

  Attributes:

  - ID

## 4.2 Description of Elements

### 4.2.1 Discourse Entities

***Description***

The assumption underlying most annotation schemes for coreference is that processing text involves building a **discourse model** containing **discourse entities**, and that anaphoric relations are relations between these discourse entities (Webber, 1978; Heim, 1982; Kamp, 1981). We use the <coref:de> tag to annotate the text spans that introduce a discourse entity - that is, that can be subsequently referred to by means of anaphoric expressions. These are commonly noun phrases. Not all noun phrases do this: for example, whereas

*John likes Bill*

introduces two discourse entities, as can be shown by the fact that a follow-up like

*He is crazy*

is ambiguous in that *he* can refer either to *John* or to *Bill*, the sentence

*John is a policeman*

which from a syntactic point of view also contains two NPs, nevertheless only introduces one discourse entity, as can be seen by the fact that in this case, the continuation *He is crazy* is not ambiguous. As a consequence, the NP *a policeman* would not get a <coref:de> tag; in other words, the textual elements given a <coref:de> tag are a subset of the range of NPs.

### *Data Source*

The annotation for <coref:de>'s should be included in a file with pointers to a base file which has already been XML tagged with information about the structure of the conversation, ideally using TEI coding (http://etext.virginia.edu/TEI.html), suitably converted into XML. A typical dialogue marked up in TEI has a <teiHeader>, <head>, and a <body> which is broken up into utterances (<u>), marked for speaker. Each <pause> is marked. The <u> might be further segmented, for example into prosodic phrases, using the TEI <seg> tags. Gestures and mouse clicks may also be marked, as may notes made by the annotator or the initial transcriber, and more detailed information can be given about pause durations, type of transitions between speakers, and many other features. The French conversation in (4.1), for example (from the Microfusées corpus), might be marked up as in (4.2):

(4.1)
Formateur: Alors donc / vous avez / ici [au milieu de la table] / les modèles des fusées volé /

[Le formateur dispose le petit paquet de dessins des 9 fusées.]

Mia: Oui

Formateur: Et vous allez essayer de vous mettre d'accord sur un classement / hein classer les fusées qui ont bien volé ou qui ont moins bien volé /

[Le formateur montre avec les mains un endroit (bien volé puis un autre (moins bien volé.]

Mia: Alors par exemple de celle qui a / le / qui a volé le plus loin / à à celle qui a volé moins loin(?)

Instructor: OK, then, here you have [in the middle of the table] the models of the rockets.

[The instructor puts down the little packet of 9 rocket designs.]

Mia: Yes

Instructor: And you are going to try to agree on a classification... to classify the rockets which flew well or which flew less well.

[The instructor points to one place (those which flew well) then another (those which flew less well)]

Mia: So for example from the one which.. it.. which flew the furthest... to the one which flew the least far?

(4.2)

```
<u id="u1" who="F">
  <seg id="u1seg1">
    Alors donc
    <pause dur="short"/>
    vous avez
    <pause dur="short"/>
    ici
    <note place="inline">
      au milieu de la table
    </note>
    <pause dur="short"/>
    les modèles des fusées
    <pause dur="short"/>
  </seg>
  <note pla e="outline" type="stage dire tions">
```

```
    Le formateur dispose le petit paquet de dessins des 9 fusées.
  </note>
</u>
<u id="u2" who="M" trans="pause">
  <seg id="u2seg1">
    Oui
  </seg>
</u>
<u id="u3" who="F">
  <seg id="u3seg1">
    Et vous allez essayer de vous mettre d'accord sur un classement
    <pause dur="short"/>
  </seg>
  <seg id="u3seg2">
    hein classer les fusées qui ont bien volé ou qui ont moins bien volé
    <pause dur="short"/>
  </seg>
  <note place="outline" type="stage directions">
    Le formateur montre avec les mains un endroit (bien volé) puis un autre
    (moins bien volé) .
  </note>
</u>
<u id="u4" who="M" trans="pause">
  <seg id="u4seg1">
    Alors par exemple de celle qui a
    <pause dur="short"/>
    le
    <pause dur="short"/>
    qui a volé le plus loin
    <pause dur="short"/>
    à celle qui a volé moins loin (?)
  </seg>
</u>
<u id="u1" who="F">
  <seg id="u1seg1">
    OK, then,
    <pause dur="short"/>
    you have
    <pause dur="short"/>
    here
    <note place="inline">
      in the middle of the table
    </note>
    <pause dur="short"/>
    the models of the rockets
    <pause dur="short"/>
  </seg>
  <note place="outline" type="stage directions"/>
    The instructor puts down the little packet of 9 rocket designs
  </note>
</u>
<u id="u2" who="M" trans="pause">
  <seg id="u2seg1">
    Yes
  </seg>
</u>
<u id="u3" who="F">
  <seg id="u3seg1">
    And you are going to try to agree on a classification
    <pause dur="short"/>
  </seg>
  <seg id="u3seg2">
    to classify the rockets which flew well or which flew less well
    <pause dur="short"/>
  </seg>
  <note place="outline" type="stage directions">
    The instructor points to one place (those which flew well) then another
    (those which flew less well).
  </note>
</u>
<u id="u4" who="M" trans="pause">
  <seg id="u4seg1">
    So for example from the one which
    <pause dur="short"/>
    it
    <pause dur="short"/>
    which flew the furthest
    <pause dur="short"/>
    to the one which flew the least far (?)
  </seg>
</u>
```

The details of the TEI mark-up may not suit all corpora, depending on the format in which the initial transcription has been presented. For example, in the TRAINS corpus each speaker turn is segmented into a number of different utterances, separated at prosodic phrase boundaries (4.3).

This means that the <u> are much shorter than those in true TEI-conformant mark-up, and there is then no TEI tag suitable for grouping the utterances into turns. For the moment, we have adopted the procedure in this case of introducing <turn> tags for a whole turn, and using <u> for each utterance or prosodic phrase:

(4.3)

```
44.1   S: +okay+
44.2    : okay
44.3    : lemme run /
44.4    : lemme make sure I got all this
44.5    : okay
44.6    : you wanna send E2
44.7    : you wanna link
44.8    : uh
44.9    : the boxcar at Elmira to E2
44.10   : and send that to Corning
45.1   M: yeah
46.1   S: and have it load oranges
47.1   M: right
48.1   S: okay
```

(4.4)

```
<turn id="t44" who="S">
  <u id="u44.1">+okay+</u>
  <u id="u44.2">okay</u>
  <u id="u44.3">lemme run</u>
  <u id="u44.4">lemme make sure I got all this</u>
  <u id="u44.5">okay</u>
  <u id="u44.6">you wanna send E2</u>
  <u id="u44.7">you wanna link</u>
  <u id="u44.8">uh</u>
  <u id="u44.9">the boxcar at Elmira to E2</u>
  <u id="u44.10">and send that to Corning</u>
</turn
<turn id="t45" who="M">
  <u id="u45.1">yeah</u>
</turn
<turn id="t46" who="S">
  <u id="u46.1">and have it load oranges</u>
</turn
<turn id="t47" who="M">
  <u id="u47.1">right</u>
</turn
<turn id="t48" who="S">
  <u id="u48.1">okay</u>
</turn>
```

If one wishes to impose syntactic restrictions on potential markables - which is a good idea for annotation exercises of any complexity - then this basic level must be further annotated with something which allows that constraint to be expressed - word tags, or full syntactic elements, or morpho-syntax tags as defined in the MATE Morpho-syntax scheme (Pirrelli and Soria, 1999). Since different schemes make different choices, the exact data source requirements are left to the individual schemes.

***Assignment***

The only attributes of `<coref:de>` that have to be set are id and href, both of which are automatically computed by the MATE workbench, either by making `<coref:de>` elements match the output of some MATE query on morphosyntactic tagging or by computation from text selected in the coding interface by the human user.

***Example***

Assuming that chunks with nominal governors are chosen as markables and that the sentence

(4.5) *John likes Bill*

would get annotated with chunks as follows:

(4.6)

ch.xml

```
<ch id="ch_001" type="N">
  <potgov id="p_001">
    John
  </potgov>
</ch>
<ch id="ch_002" type="V">
  <potgov id="p_002">
    likes
  </potgov>
</ch>
<ch id="ch_003" type="N">
  <potgov id="p_003">
    Bill
  </potgov>
</ch>
```

then the following discourse entities would be annotated:

(4.7)

coref.xml

```
<coref:de id="de_001" href="ch.xml#id(ch_001)"/>
<coref:de id="de_002" href="ch.xml#id(ch_003)"/>
```

*Important Note*: Since the underlying XML representation is meant to be transparent to the annotator using the MATE tools, in the examples below we have simplified the notation considerably so as to make it easier for non-XML experts to understand the annotation; this would also make it clearer that the meta-scheme does not crucially depend on a particular type of basic level markup. First of all, we give examples in plain text, abstracting away from the chunking level, except in a few cases when this is necessary. Second, instead of representing the markup by means of href pointers as in (4.7), we will adopt a more conventional SGML-style format with tags wrapped around the parts of the text to be annotated with a `<coref:de>` element, so as to make it clearer to the annotator which part of the text to highlight and to mark; the representation in (4.7) will be automatically constructed by the tool and the annotator need not be aware of it. In our examples, we will generally use the following representation, rather than the format in (4.7):

(4.8)

```
<coref:de>John</coref:de>
  likes
<coref:de>Bill</coref:de>
```

***Coding Procedure***

Left to the individual schemes.

***Markup Table***

| <coref:de> | |
|---|---|
| id | [ASCII] |
| href | <ch> |

### 4.2.2 Link and Anchor Entities

***Description***

<coref:link> elements are used to mark anaphoric relations between discourse entities, the most basic of which is the identity relation. This relation obtains between two phrases in a text when they denote the same object in the world; the phrases used to refer to this object can be the same, like *'la surface... la surface'* in (4.9), *'orange juice... orange juice'* in (4.10), *'les ailerons... les ailerons'* in (4.11) or different, as is seen with *'the engine E3... it... it'* in (4.12), or *'ces deux fusées... elles'* in (4.13). As these last two examples suggest, it is very common for a pronoun to be used to refer to a discourse entity previously referred to by a full noun phrase.

(4.9)

```
S: Créer la surface.
W: Opération effectuée
S: Modéliser la surface
W: Quel nom voulez-vous donner à la surface ?
S: Create the surface
W: Done
S: Model the surface
W: What name do you want to give to the surface ? (MF)
```

(4.10)

```
When do we have orange juice at Elmira?
We have orange juice at Elmira at 6 a.m. (T)
```

(4.11)

```
197 F: mmh / Donc qu'est ce que vous allez garder en fait (?) + /
198 M: |la longueur du tube et les ailerons |
199 D: |les ailerons |
200 F: Donc les ailerons vous m'avez dit.
197 F: mmm / Well, what are you going to keep, then ? /
198 M: the length of the tube and the wings |
199 D: |the wings |
200 F: well, the wings, you said (MF)
```

(4.12)

```
we're gonna take the engine E3 and shove it over to Corning, hook it up to the tanker car... (T)
```

(4.13)

193 F: Donc qu'est ce qui / qu'est ce qui serait commun à *ces deux fusées*. *Ces deux fusées* ont /
194 D: c'est qu'*elles* ont / *elles* ont la même...
193 F: What would it be that these two rockets have in common? These two rockets have /
194 D: it's that they have / they have the same... (MF)

*A group of children* perform an intricate dance in a small theatre in the northern Sri Lankan town of Jaffna.
The appreciative audience sit in the open air and applaud *their* performance.
The members of the Centre for Performing Arts in Jaffna are justly proud of *their* performance...(BBC)

In this section we only discuss the case of links describing identity relations, but nothing prevents an annotator to use a wider range of relations, as done in the DRAMA scheme; some suggestions concerning possible relations are in Section 8.

### *Data Source*

The <coref:link> and <coref:anchor> elements point to <coref:de> elements.

### *Segmentation/Selection*

Not applicable (the information provided by <coref:link> elements comes entirely from their attributes).

### *Assignment*

The HREF attributes of link and anchor elements both refer to the ID of an antecedent, which can be either a <coref:de> element, a <coref:ue> element, or a <coref:seg> element (see below). For the moment, we assume that the antecedent denotes the same object as the <coref:de> element, and the ident relation is used. We assume in the rest of this document that the annotation is contained within a file 'coref.xml' to which the href elements point.

*Coreference chains*: It is often the case that more than two discourse entities refer to the same object; in this case, a **coreference chain** is formed. Because the identity relation is transitive, if A is ident with B and B is ident with C, then A is ident with C; so it doesn't matter which item in a coreference chain is chosen as antecedent for a new phrase. This can be tracked through the markup.

Furthermore, since the identity relation is symmetric, it doesn't matter which <coref:de> element is chosen as 'current element' and which one as 'anchor'. It is often less confusing, however, to adopt the convention that the <coref:link> element should point to the latest discourse entity, whereas the <coref:anchor> element should point to the antecedent.

*Participants interpret anaphoric expressions differently*: It is also possible to observe that at a certain point in a dialogue the conversational participants had differences of opinion about coreferential links. For this reason, links can contain specifications of which agent or set of agents believes them to hold, via the optional WHO-BELIEVES attribute. The default value for this attribute is SHARED.

### *Example*

We use the <coref:link> and <coref:anchor> elements to mark anaphoric relations, as follows. When two noun phrases marked as <coref:de> elements co-specify, a <coref:link> element is added. The href attribute of this element points to the anaphoric expression, and contains at least one <coref:anchor> element specifying the antecedent (by means of a second href pointer). The type of relation that holds between the two discourse entities (the values of which depend on the exact scheme implemented) is specified by the type attribute of the <coref:link> element. (As we will see below, specifying anaphoric relations by means of elements embedded into a <coref:link> element allows the annotator to mark for ambiguities of co-specification.) Here are some example annotations.

(4.15)

coref.xml

```
When do we have<coref:de ID="de _01">orange juice</coref:de>at Elmira?
We have <coref:de ID="de _02">orange juice</coref:de>at Elmira at 6 a.m. (T)
<coref:link type="ident" href="coref.xml#id(de_02)">
  <coref:anchor href="coref.xml#id(de_01)"/>
</coref:link>
```

(4.16)

coref.xml

```
197 F: mmh / Donc qu'est ce que vous allez garder en fait (?) + /
198 M: |la longueur du tube et <coref:de ID="de _98">les ailerons</coref:de>
199 D:<coref:de ID="de_99">les ailerons</coref:de>
200 F: Donc <coref:de ID="de_100">les ailerons</coref:de> vous m'avez dit.
<coref:link href="coref.xml#id(de_99)" type="ident">
  <coref:anchor href="coref.xml#id(de_98)" />
</coref:link>
<coref:link href="coref.xml#id(de_100)" type="ident" >
  <coref:anchor href="coref.xml#id(de_99)"/>
</coref:link>
```

(4.17)

```
we're gonna take <coref:de ID="de_07">the engine E3</coref:de> and shove
<coref:de ID="de_08">it</coref:de> over to Corning, hook
<coref:de ID="de_09">it</coref:de> up to the tanker car...
<coref:link href="coref.xml#id(de_08)" type="ident">
  <coref:anchor href="coref.xml#id(de_07)"/>
</coref:link>
<coref:link href="coref.xml#id(de_09)" type="ident">
  <coref:anchor href="coref.xml#id(de_08)"/>
</coref:link>
```

*Ambiguity*: The reason why more than one <coref:anchor> element may be embedded in a <coref:link> element is to annotate ambiguity. In case more than one entity appear to be equally likely antecedents for an anaphoric expression, each of the possibilities can be marked by means of a separate <coref:anchor> element. In the following example, the pronoun it in 15.16 could refer equally well to engine E3 or to the tanker car. If the annotator desires to annotate both antecedents, as in DRAMA or in the Lancaster scheme, this can be done as shown below.

coref.xml

```
15.12 : we're gonna take <coref:de ID="de_15">the engine E3</coref:de>
15.13 : and shove <coref:de ID="de_16">it</coref:de> over to Corning
```

```
15.14 : hook <coref:de ID="de_17">it</coref:de> up to
        <coref:de ID="de_18">the tanker car</coref:de>
15.15 : _and_
15.16 : and send <coref:de ID="de_19">it</coref:de> back to Elmira

<coref:link href="coref.xml#id(de_16)" type="ident">
  <coref:anchor href="coref.xml#id(de_15)"/>
</coref:link>
<coref:link href="coref.xml#id(de_17)" type="ident">
  <coref:anchor href="coref.xml#id(de_16)"/>
</coref:link>
<coref:link href="coref.xml#id(de_19)" type="ident">
  <coref:anchor href="coref.xml#id(de_17)"/>
  <coref:anchor href="coref.xml#id(de_18)"/>
</coref:link>
```

***Coding Procedure***

Left to the individual schemes.

***Markup Table***

| <coref:link> | |
|---|---|
| id | [ASCII] |
| who-believes | [ASCII] |
| type | ident, member, subset, poss, e-rel, argptv, prop, bound, f-v, inst, genrel |
| subtype | attr, part, sposs, cause |
| href | <coref:de> |
| content | <coref:anchor> |

| <coref:anchor> | |
|---|---|
| id | [ASCII] |
| href | <coref:de> |

### 4.2.3 Universe and UE Entities

In face-to-face or human-machine dialogue, participants may make reference to items visible to them at the time of speaking. A simple example of this is *Pass the salt, please*, where *salt* may not have been previously mentioned in the conversation, and thus does not corefer with any other `<coref:de>`, but does refer to an entity which is in the visible situation. Tracking these references is important for multimodal systems (Bruneseaux and Romary, 1997), and they have been annotated reliably in the MapTask. This tracking requires two new elements: a `<coref:universe>` element (as in the Bruneseaux and Romary scheme) used to specify a 'universe of discourse', that is, a set of objects, each specified by a `<coref:ue>` element.

The <coref:universe> element may also be used to specify references to items in the non-visible 'universe' of shared knowledge which allows hearers to correctly assign reference to items such as *the Eiffel Tower* - the so-called 'larger-situation' (Hawkins, 1978) or 'hearer-old' (Prince, 1981) references; however, annotators should keep in mind that it is often difficult to do such categorizations reliably, as found out by Fraurud (1990) and Poesio and Vieira (1998).

### *Description*

In order to mark up reference to items in the visual situation, the items in the visual situation are listed as **universe entities (<**coref:ue**>)**, embedded within a <coref:universe> element. Each <coref:ue> element has an ID, like <coref:de> do, so that a relation of identity between a noun phrase and an object in the visual situation can be encoded by an ident link between a <coref:de> and a <coref:ue> just like identity between two <coref:de> elements.

Where feasible, it is suggested that all objects in the visual situation be included in a single <coref:universe> element. In cases like the MapTask dialogues where the participants to the conversation have two different maps, it is suggested that three universes be created: one with ID common containing all objects shared between the visual situations, and then one universe for each conversational participant containing the elements known only to that element, and with value modifies="common". This will ensure that the shared elements receive a unique ID.

In some types of dialogues the visual situation may change: new objects may be created and old objects destroyed (e.g.,when the visual situation is the screen). These situations may be modeled by allowing for the creation of new universes in the middle of dialogues, although this is not yet supported.

### *Data Source*

There are no additional requirements on source data for the use of universes, unless a scheme implements a restriction on what coreferences are to be annotated based on the types of objects referred to; in this case, the annotator needs a description of the objects to check against. For instance, if the annotator were to mark up only references to Map Task landmarks, then the annotator would need a list of landmarks or copies of the maps. This information may not be enshrined in the data files themselves but in the coding module for the scheme instantiation.

### *Segmentation*

Not applicable.

### *Assignment*

The modifies attribute for all but the common universe should be set to common.

### *Example*

The following is a simple example of the use of a universe.

(4.18)

```
<coref:universe ID="u1">
  <coref:ue ID="ue1">Diamond mine</coref:ue>
  <coref:ue ID="ue2">Graveyard</coref:ue>
  <coref:ue ID="ue3">Fast running creek</coref:ue>
  <coref:ue ID="ue4">Fast flowing river</coref:ue>
  <coref:ue ID="ue5">Canoes</coref:ue>
</coref:universe>

FOLLOWER: Uh-huh. Curve round. To your right.
GIVER:    Uh-huh.
FOLLOWER: Right.... Right underneath <coref:de ID="de_50">the diamond mine.</coref:de>
          Where do I stop.
GIVER:    Well..... Do. Have you got <coref:de ID="de_51">a graveyard?</coref:de>
          Sort of in the middle of the page? ... On on a level to
          <coref:de ID="de _52">the c-- ... er diamond mine.</coref:de>
FOLLOWER: No. I've got <coref:de ID="de_53">a fast running creek.</coref:
GIVER:    <coref:de ID="de _54">A fast flowing river</coref:de>,... eh.
FOLLOWER: No. Where's <coref:de ID="de_55"> that </coref:de>. Mmhmm,... eh.
          <coref:de ID="de _56">Canoes</coref:de>
<coref:link href="coref.xml#id(de_50)" type="ident">
  <coref:anchor href="coref.xml#id(ue1)"/>
</coref:link>
<coref:link href="coref.xml#id(de_51)" type="ident">
  <coref:anchor href="coref.xml#id(ue2)"/>
</coref:link>
<coref:link href="coref.xml#id(de_52)" type="ident">
  <coref:anchor href="coref.xml#id(ue1)"/>
</coref:link>
<coref:link href="coref.xml#id(de_53)" type="ident">
  <coref:anchor href="coref.xml#id(ue3)"/>
</coref:link>
<coref:link href="coref.xml#id(de_54)" type="ident">
  <coref:anchor href="coref.xml#id(ue4)"/>
</coref:link>
<coref:link href="coref.xml#id(de_55)" type="ident">
  <coref:anchor href="coref.xml#id(de_54)"/>
</coref:link>
<coref:link href="coref.xml#id(de_56)" type="ident">
  <coref:anchor href="coref.xml#id(ue5)"/>
</coref:link>
```

Note that `<coref:de ID="de_55">`, *that*, could be marked up as ident with either the universe entity ue4, or with the discourse entity de_54. One of the advantages of this way of annotating references to the visual situation is that an extended coreference chain tracking mechanism should be able to include in a coreference chain both references to universe elements and references to discourse entities; the annotator may then choose how he/she wishes to annotate this. If the annotation tool can't do this type of coreference chain tracking, then the coding manual should include a disambiguation rule: for the type of multimodal applications on which Bruneseaux and Romary worked it seems preferable to mark links with universe entities rather than marking links with previous discourse entities.

The following is a more complex example which includes multiple universes encoded different world knowledge and a disagreement about a coreferential link in the dialogue.

(4.19)

```
<coref:universe ID="common">
  <coref:ue ID="ue2">gold mine</coref:ue>
</coref:universe>
<coref:universe ID="GIVER_universe" modifies="common">
  <coref:ue ID="ue1">diamond mine</coref:ue>
</coref:universe>
<coref:universe ID="FOLLOWER_universe" modifies="common">
.....
</coref:universe>

GIVER:    Do_you have <coref:de ID="de_20">diamond_mine.</coref:de>
FOLLOWER: Yes I've got <coref:de ID="de_21">a gold mine.</coref:de>
```

```
GIVER:    Ah. S--.
FOLLOWER: ....
GIVER:    You don't have <coref:de ID="de_22">diamond_mine</coref:de> though.
FOLLOWER: No. It's <coref:de ID="de_23"> a gold_mine </coref:de> according to this one.
          Presumably <coref:de ID="de_24">that's</coref:de> the same.
GIVER:    Well I've got <coref:de ID="de_25">a gold_mine</coref:de> as well you see.
<coref:link href="coref.xml#id(de_20)" who-believes="G" type="ident">
  <coref:anchor href="coref.xml#id(ue1)"/>
</coref:link>
<coref:link href="coref.xml#id(de_21)" who-believes="F" type="ident">
  <coref:anchor href="coref.xml#id(ue2)"/>
</coref:link>
<coref:link href="coref.xml#id(de_21)" who-believes="F" type="ident">
  <coref:anchor href="coref.xml#id(de_20)"/>
</coref:link>
<coref:link href="coref.xml#id(de_22)" who-believes="G" type="ident">
  <coref:anchor href="coref.xml#id(ue1)"/>
</coref:link>
<coref:link href="coref.xml#id(de_22)" type="ident">
  <coref:anchor href="coref.xml#id(de_20)"/>
</coref:link>
<coref:link href="coref.xml#id(de_23)" who-believes="F" type="ident">
  <coref:anchor href="coref.xml#id(ue2)"/>
</coref:link>
<coref:link href="coref.xml#id(de_23)" who-believes="F" type="ident">
  <coref:anchor href="coref.xml#id(de_22)"/>
</coref:link>
<coref:link href="coref.xml#id(de_24)" who-believes="F" type="ident">
  <coref:anchor href="coref.xml#id(de_22)"/>
</coref:link>
```

### *Coding Procedure*

The annotation should begin with the creation of a `<coref:universe>` element (or a common universe plus one for each participant, if their knowledge is not the same). This is commonly done before the annotation of discourse entities if the universe is static.

### *Markup Table*

| <coref:universe> | |
|---|---|
| id | [ASCII] |
| modifies | <ch> |
| content | <coref:ue> |

| <coref:ue> | |
|---|---|
| id | [ASCII] |
| content | TEXT: description of object |

## 4.2.4 Seg Elements

### *Description*

Even if we only consider anaphoric relations involving nominal elements, there are at least two situations in which an annotator may wish to mark an anaphoric relation that also involves other

types of constituents. The first is the case in which the anaphoric element is either unexpressed or incorporated in the verb. The second situation are the cases of so-called **discourse deixis** (Webber, 1991), in which the antecedent of a nominal expression is an abstract object such as an event or proposition introduced in the discourse somewhat indirectly by sentences. (DRAMA allows for such relations to be marked.)

The solution we propose is to use a <coref:seg> element which, like the TEI `<seg>` element, can be used to mark up arbitrary pieces of text. `<coref:seg>` elements are given an id which can then be pointed at by a `<coref:link>` element just like for other anaphoric relations.

The `<coref:seg>` element could also be used to annotate anaphoric relations between non-nominal elements, such as in VP ellipsis.

### *Data Source*

Data source requirements for `<coref:seg>` elements are the same as for `<coref:de>` elements.

### *Segmentation*

To be specified by the coding manual for a given scheme.

### *Assignment*

The id attribute is automatically set by the workbench.

### *Example*

**Using `<coref:seg>` to mark up empty and incorporated constituents**: As seen above, in Italian, Spanish and many other languages, certain nominal constituents may not be realized; this is especially common for nominals in subject position, but can also happen in object position, especially in instructions, as in:

*Add the dry yeast to the water and let _ sit for a few minutes. Add the rest of the water and sugar. Stir _*

These nominals are present in annotations produced by hand (e.g., in the Penn Treebank), but the parsers used for parsing spoken dialogues tend not to produce representations containing empty constituents in this case. In case these nominals are not represented in the base level, we verb can be marked with a `<coref:seg>` element, and the anaphoric relation coded as usual by means of `<coref:link>` elements, as follows:

(4.20)

coref.xml

```
A: Dov'e' <coref:de ID="de_157">Gianni?</coref:de>
   [Where is Gianni?]
B: <coref:seg type="pred" ID="seg_158>
```

```
    e‘ andato a mangiare
  </coref:seg>
  [_ went to have lunch]
<coref:link href="coref.xml#id(seg_158)" type="ident">
      <coref:anchor      href="coref.xml#id(de_157)"/>
</coref:link>
```

This representation can only be used without loss of information when there is at most one empty elements; this is true for Italian, but not for Japanese or Portuguese. If more precision is needed, the annotator could define more specific identity relations also specifying which empty argument of the verb enters in the anaphoric relation: such relations could be called, e.g., subj-ident, obj-ident, etc. These relations could then used instead of ident as the value of the type attribute of the <coref:link> element; we won't make them part of the annotation scheme discussed here, however.

A second case in which an argument is not realized by means of a nominal is that of incorporated clitics, such as *daselo* in (4.21) below. Clitic suffixes are also found in transcriptions of spoken English:

```
44.4 : lemme make sure I got all this
44.5 : okay (T)
```

In the case of incorporated clitics, as well, the verb can be marked with a <coref:seg> element when the parser doesn't produce a morphologically decomposed representation, and then the anaphoric relations in which the clitics are involved can be encoded either by means of a single ident relation or by means of more fine-grained relations such as subj-ident or obj-ident.

(4.21)

coref.xml

```
A: Mira, te doy <coref:de ID="de_167">este libro</coref:de>
   Â¿Conoces a <coref:de ID="de_168">mi suegra?</coref:de>
B: Sí, claro.
A: Pues <coref:seg ID="seg_169">dáselo</coref:seg> cuando
   <coref:de ID="de_170">la</coref:de> veas.
<coref:link href="coref.xml#id(seg_169)" type="obj-ident">
  <coref:anchor href="coref.xml#id(de_167)"/>
</coref:link>
<coref:link href="coref.xml#id(seg_169) type="iobj-ident">
  <coref:anchor href="coref.xml#id(de_168)"/>
</coref:link>
```

Provided that the `<coref:seg>` elements are identified during the first pass of markable identification, encoding this information should not be any harder than in the case of MUCCS. The real question for this type of annotation is which empty elements to annotate --e.g., in addition to 'small pro' elements such as those discussed above, the annotator may also decide to annotate 'big PRO' elements that according to some syntactic theories occupy the subject position of infinitival clauses.

*Using SEG to mark the antecedents of discourse deixis*: Abstract objects such as events, actions and propositions can all serve as antecedents of anaphoric expressions. We are not aware of any reliability results for this type of annotation, but the `<coref:seg>` element can be used to identify the antecedents in this type of anaphora. If desired, the annotator could use a second attribute type

to specify the type of object introduced by the <coref:seg> element; type would have values event, prop and action.

(4.22)

```
<coref:seg type="event" ID="seg_130">
  The 23-year-old had hit his head against another player
</coref:seg> during a game of Aussie-rules football.
McGlinn remembered nothing of
<coref:de ID="de_131">
  the collision
</coref:de>
, but developed a headache and had several seizures.
<coref:link href="coref.xml#id(de_131)" type="ident">
  <coref:anchor href="coref.xml#id(seg_130)"/>
</coref:link>
```

(4.23)

```
a. Despite the latest negative results, doctors are still
   convinced that Tamoxifen can prevent breast cancer.
   This is because of the way it blocks the action of oestrogen,
   the female sex hormone that can make the breast cells of some
   women go out of control.

b. Despite the latest negative results,
   <coref:seg type="prop" ID="seg_129">
   doctors are still convinced that
   <coref:de ID="de_131">Tamoxifen</coref:de>
   can prevent breast cancer
   </coref:seg>.
   <coref:de ID="de_130">This</coref:de>
   is because of the way
   <coref:de ID="de_132">it</coref:de>
   blocks the action of oestrogen, the female sex hormone that
   can make the breast cells of some women go out of control.

<coref:link href="coref.xml#id(de_130)" type="ident">
  <coref:anchor href="coref.xml#id(seg_129)"/>
</coref:link>
```

(4.24)

```
a.
GIVER:     You're sort_of going past stone creek ...
           but your line's curving up past the ... flat rocks.
FOLLOWER:  Right. Okay.
GIVER:     and then starting to come down again.
FOLLOWER:  Got that

b.
GIVER:     You're sort_of going past stone creek ...
           but your line's curving up past the ... flat rocks.
FOLLOWER:  Right. Okay.
GIVER:     <coref:seg ID="seg_135" type="action">
           And then starting to come down again.
           </coref:seg>
FOLLOWER:  Got <coref:de ID="de_136">that</coref:de>.

<coref:link href="coref.xml#id(de_136)" type="ident">
  <coref:anchor href="coref.xml#id(seg_135)"/>
/coref:link>
```

These examples also ilustrate some of the problems to be addressed when designing a reliable annotation scheme for discourse deixis: these include deciding what part of the text counts as antecedent as well as deciding which type of object the antecedent is (see, e.g., (4.24)).

***Coding Procedure***

Left to the individual schemes.

***Markup Table***

| <coref:seg> | |
|---|---|
| id | [ASCII] |
| type | [ASCII] |

### 4.3 Integrated Example

See (4.18) and (4.19).

### 4.4 Joint Coding Procedure

Left to the individual schemes.

# 5 A MUC 7-LIKE SCHEME

As a first example of how the markup elements introduced in the previous section can be used to annotate according to the indications of some of the current schemes, we show how to use them to annotate according to the MUCCS scheme developed for MUC-7 which, as mentioned above, is the simplest and most reliable of the existing schemes.

### 5.1 Markup Declaration

Two of the elements introduced in the previous section are needed for this scheme: `<coref:de>` and `<coref:link>`.

### 5.2 Description of Elements

As in the common section, except that the MUC-7 instructions should be used to segment discourse entities. Note that the MUCCS instructions also prescribe the mark up of parts of NPs, not only of full NPs. (See section 8 for a discussion of the options for `<coref:de>` markup.)

### 5.3 Integrated Example

See examples (4.15)-(4.17).

## 5.4 Joint Coding Procedure

According to the MUCCS instructions: first mark up all text elements specified as markables, then annotate all the coreference relations.

# 6 A Maptask Schema

This scheme illustrates how to fill in the modular components in order to instantiate the landmark coding which HCRC uses for the Map Task. In their original work, no syntactic constraints on markables were applied, since for their purposes the syntactic form of referring expressions was an empirical question, but annotation was limited to references to Map Task landmarks. This scheme requires the annotator to set up universes of landmarks corresponding to the set of landmarks on both maps, the set of objects only on the giver map, and the set of objects only on the follower map.

## 6.1 Markup Declaration

Mark-up is as in the common section.

## 6.2 Description of Elements

As in the common section, with the following exceptions:

### 6.2.1 Discourse Entities

***Description***

The `<coref:de>` tag is used to mark only spans of words referring to a landmark.

***Data Source***

Since annotators mark spans of words which refer to landmarks, the data must be marked up with words which have IDs, since thee will be used to fill in HREF attributes of the DE elements.

***Segmentation/Selection***

Annotators should only select text spans which refer to landmarks. Selection requires the annotator to have access either to a list of landmarks or to the Giver and Follower maps.

***Assignment***

The TYPE of DE elements must be set to IDENT.

***Example***

See examples 4.17 and 4.18, under "Common Markup".

***Coding Procedure***

See "Joint Coding Procedure" below.

### 6.2.2 Universes and universe elements

***Description***

All common landmarks should be included in a universe called common; then all landmarks in the giver's map only should be included in a universe called GIVER_universe, whereas all the landmarks included in the follower's map only should be included in a universe called FOLLOWER_universe.

***Data Source***

The annotation of these elements is done on the basis of the maps or a landmark list.

***Assignment***

The labels of the landmarks in the map should be used as the content of the coref:ue elements. The modifies attribute of the GIVER_universe and FOLLOWER_universe coref:universe elements should be set to common.

***Example***

See examples 4.17 and 4.18, under "Common Markup".

***Coding Procedure***

See "Joint Coding Procedure" below.

### 6.2.3 Links

***Description***

The `<coref:link>` element is used to annotate references to the landmarks.

***Data Source***

The `<coref:link>` elements point to `<coref:de>` elements and `<coref:ue>` elements.

***Segmentation/selection***

A `<coref:link>` should be specified for every `<coref:de>` element.

***Assignment***

The href attribute of the `<coref:link>` element should be set to a discourse entity; the href attribute of the `<coref:anchor>` should be set to a universe entity. The WHO-BELIEVES attribute can be used to annotate misunderstandings, and in cases of ambiguity, two anchor elements may be included.

***Example***

See examples 4.17 and 4.18, under "Common Markup".

***Coding Procedure***

See "Joint Coding Procedure" below.

### 6.3 Integrated example

See examples 4.17 and 4.18, under "Common Markup".

### 6.4 Joint Coding Procedure

The annotator, after having set up the universes of landmarks, reads the dialogue from start to finish, looking for references to landmarks to mark as discourse entities and simultaneously linking them to universe elements.

## 7 Drama

DRAMA (Passonneau, 1996) can also be implemented using the markup elements above, but extending the range of values of the TYPE attribute.

### 7.1 Markup Declaration

As in the common parts of the scheme.

### 7.2 Description of Elements

As in the common section, with the exception that the TYPE attribute of `<coref:link>` elements may be one of: {coref, subset, member, part, cause, poss, argptv, prop}. No restriction should be placed on the types of objects for which coreference information is specified. DRAMA gives explicit instructions about the syntax of markables which must be realized either using a MATE query on the morphosyntactic tagging or by the human coder.

### 7.3 Integrated Example

See Passonneau's manual.

### 7.4 Joint Coding Procedure

See Passonneau's manual.

## 8 An Overview of the Possible Decision about Markup

In this section we discuss in more detail some of the issues that the designer of a coreference scheme must confront, and give some suggestions.

## 8.1 Marking Up Discourse Entities

This section offers guidance on the types of words and phrases which may be marked up as discourse entities (<coref:de>). As in MUCCS and in DRAMA, we only give instructions about how to mark possible antecedents for nominal anaphora; not about marking antecedents of verbal ellipsis or other forms of anaphoric relations. The discussion follows roughly the order of the analogous discussion in DRAMA.

Our intention was to cover as many types of possible antecedents as possible, compatibly with what we think can be annotated reliably; depending on the particular application, only a subset of the markables identified here may actually be marked. The most important decision to be made by the designer of an 'instance' of the present scheme is whether to annotate all NPs, or only those that enter into anaphoric relations (i.e., are either anaphoric expressions or the antecedents of anaphoric expressions). In what follows, we assume that all NPs introducing discourse entities have to be specified as <coref:de> elements, and discuss which ones do not introduce discourse entities.

Whatever the decision is made concerning the text constituents to mark, the experience with MUC indicates that it's best to split the annotation task in two - reaching agreement on a set of <coref:de>s before attempting to specify the anaphoric relations with <coref:link> elements as discussed in the next section.

### 8.1.1 NPs with head noun

The 'canonical 'noun phrase consists of a head noun optionally pre- or post-modified by determiners, quantifiers, adjectives, etc. The whole NP (not just the head noun) is marked up wherever it denotes an entity which may be subsequently or previously referred to elsewhere in the text. Both definite and indefinite NPs can potentially enter into this kind of relationship. The following examples show some examples of canonical NPs which would be marked up (only the markup for the first example is shown).

```
(8.1) France came from behind to beat Croatia 2-1 and reach
      their first World Cup final at the Stade de France. (BBC)
(8.2) France came from behind to beat Croatia 2-1 and reach
      <coref:de id="de_30">their first World Cup final</coref:de>
      at the Stade de France. (BBC)
(8.3) But home fans at the Stade de France endured an agonising
      final 20 minutes after Laurent Blanc was shown the red card
      following a tussle with Slaven Bilic. (BBC)
(8.4) Prolific Davor Suker gave the keeper no chance for
      his fifth goal of the tournament. (BBC)
(8.5) A high-class move followed two minutes later with Youri Djorkaeff
      finding Guivarc'h with an excellent ball. (BBC)
(8.6) The first three students came in.
(8.7) A lot of students followed
```

Note that in the case of *the first three students*, *three students* is not marked separately even though it could occur by itself in NP position; this is because the subconstituent could not serve as antecedent by itself. However, where the quantifier occurs in an *of*-construction with a full NP complement, both NPs should be marked as both could serve as antecedents, as follows:

```
(8.8) Some of the symptoms are barely noticeable except when the patient is tired.
(8.9) <coref:de ID="DE_101">
        Some of
        <coref:de ID="DE_102">
          the symptoms
        </coref:de>
      </coref:de>
      are barely noticeable except when the patient is tired.
```

NPs should only be marked up where they do introduce a new discourse entity. As we will see below, there are cases in which this is clearly not the case: e.g., when an indefinite NP occurs as a predicate nominal. In some cases it is difficult to tell whether a given indefinite NP is predicative or not. Our suggestion is to keep the rules for these cases simple: unless it has been decided only to mark NPs that actually enter into anaphoric relations, mark all indefinite NPs as `<coref:de>`s except when they occur in the special constructions discussed in 8.1.8.

It should be noticed that indefinite NPs may introduce discourse entities even whey they do not refer to anything in the world: e.g., in

(8.10)

```
I want to buy a car.
```

*A car* does not refer to any particular object in the world (unlike, say, in *I've just seen a lovely car, but it was too expensive*). Yet, that car can still occur in anaphoric relations, although under particular conditions ('modal subordination'): e.g., it is possible to continue (8.10) by saying *I need it/one to go to work.* (Indeed, this possibility of anaphoric relations to expressions that do not refer is the reason why the intermediate level of 'discourse entity' was introduced - see e.g., (Karttunen, 1967) or (Webber, 1978).)

### 8.1.2 NPs containing relative clauses

Where an NP contains a **restrictive** relative clause, the whole NP, including the restrictive relative clause, should be marked up as single discourse entity.

```
(8.11) They will play Brazil on Sunday after a dominant performance against
       a Croatia side who surprised many by reaching the semi-final stage. (BBC)
(8.12) They will play Brazil on Sunday after a dominant performance against
       <coref:de ID="DE_31">
         a Croatia side who surprised many by reaching the semi-final stage
       </coref:de>.
```

It may be argued that **non-restrictive** relative clauses should **not** be marked up as part of the discourse entity, as they supply additional information about the referent rather than helping to identify him or her. However, it may be useful for the goal of the annotation to include information about the pattern of reference of non-restrictive relative clauses, and annotators may therefore decide that these elements should be marked up. (If the non-restrictive relative clause is not to be marked up, the `<coref:de>` tag should be assigned to the part of the NP that precedes the relative clause.) (8.14) illustrates how to tag a non-restrictive relative clause leaving it outside the `<coref:de>` tag; (8.15) how to include it.

```
(8.13) The 26-year-old Thuram, who had never before scored for France,
       scored twice after prolific Davor Suker had put Croatia in front
       at the start of the second half. (BBC)
(8.14) <coref:de ID="DE_32">
         The 26-year-old Thuram
       </coref:de>
       , who had never before scored for France, scored twice after prolific
       Davor Suker had put Croatia in front at the start of the second half.
(8.15) <coref:de ID="DE_33">
         The 26-year-old Thuram,
         who had never before scored for France
       </coref:de>
       , scored twice after prolific Davor Suker had put Croatia in front at
       the start of the second half.
(8.16) A spokeswoman for the Prince of Wales has confirmed that the encounter,
       which is said to have been amicable, did take place, but stressed it was
       a private matter for the family. (BBC)
       We assume in what follows that non-restrictive relative clauses, as well,
       are included inside the <coref:de tag for a given NP.
```

### 8.1.3 Bare nouns

Where the NP consists only of a noun, this should be marked up normally as a `<coref:de>`, as in (8.17).

(8.17)

```
5.1  M: and there're <coref:de>oranges</coref:de> at Corning (T)
```

However, bare nouns in non-head position - e.g., the premodifier *orange* in (8.18) - should normally not be marked up, since generally they do not enter in anaphoric relations; only the NP *orange juice* as a whole would be a markable.

(8.18)

```
7.2   : we have to make <coref:de>orange juice</coref:de>
```

However, the designer of a scheme may decide to allow for bare nouns in this position to optionally be marked as <coref:de>s when entering into anaphoric relations with other NPs.

Bare NPs may often be used to talk about kinds (Carlson, 1977) rather than tokens, as in utterance 1.6 in (8.19): in this example, no specific bananas have been chosen, and any will do.

(8.19)

```
1.1  M: okay
1.2   : I have to get
1.3   : one tanker of OJ
1.4   : to
1.5   : Avon
1.6   : and a boxcar of bananas
1.7   : ... to
1.8   : ... Corning
      : ...
3.2   : so there're
3.3   : bananas at Avon
```

The relation between kinds such as the one in 1.6 and sets of objects such as the one denoted by the bare NPs *bananas* in the following utterance 3.3 is not really identity of reference; at most, it can be said that the bananas in 3.3 are an instance of the type of objects mentioned in 1.6. The core scheme makes no explicit provision for this type of relation; however, if the designer of a scheme wished to mark relations like these within the core scheme, the ident relation could be used in a looser sense (denoting 'sense identity'), as done in MUCSS. The extended scheme does make provision for the mark-up of the relationship between a kind and token or instantiation of that kind; see Instantiation.

### 8.1.4 Noun phrases without a head noun: pronouns

Personal pronouns, demonstrative pronouns, possessive pronouns and indefinite pronouns may all enter into coreference and should be marked up wherever necessary.

(8.20)

```
31.4  : now I have a good i /
31.5  : oh no we can't use the same thing (T)
```

(8.21)

```
104.11   : the boxcar has a bad wheel
104.12   : and won't be available for 8 hours
105.1   M: oooh...
106.1   S: so we can't use that (T)
```

(8.22)

```
144.1   S: we get our OJ ready by 8 (T)
```

(8.23)

```
Sujet:    Euh, peut-on construire la partie supérieure
          sphérique du, de la surface ?
Compere:  Je ne dispose pas de surface
Compere:  A partir de quoi voulez-vous en créer une ? (G)
```

It is not usually necessary to mark up each occurrence of first and second person pronouns, as the cases in which they co-specify can generally be automatically determined. However, where assigning reference to these pronouns seems unusually complicated, as is sometimes the case where there are many speakers, the annotator may choose to mark them up.

Pleonastic NPs (expletives) should not be marked up, as they never enter into anaphoric relations.

(8.24)

```
85.6  : now how long does it take from Elmira to Corning? (T)
```

(8.25)

```
It seems to me that John is going mad.
```

Reflexive pronouns may be marked up as <coref:de> if they are considered to truly denote an item in the world (8.26). This should be decided on the basis of whether they identify a person or thing which could be considered an argument of the verb. For example, in the English phrases *Julie washed herself* or *Bill was talking to himself*, the reflexive pronouns clearly identify an argument of the verb, and could indeed be replaced by other noun phrases. In Spanish, however, many verbs take what might be considered 'lexicalised' reflexive pronouns: they do not appear to refer to an argument, and could not be replaced by another NP, e.g. *reirse*, to laugh, *irse* to leave/ be off. There are also cases in Spanish of the reflexive pronoun 'se' being used in impersonal and passive constructions, in which the pronoun does not seem to refer to any argument of the verb. A rough guideline for this might be that these reflexive pronouns should be marked up when they seem to coincide with their referential use in English. In (8.27), for example, the first two instances of the reflexive pronoun *se* do not seem to refer to anything, but the last (*adapts itself*) would be deemed referential and marked up as a <coref:de>. However, this may be an overly anglocentric view, and annotators may choose to adopt their own decision mechanisms in this area.

(8.26)

He picked *himself* up off the floor.

(8.27)

Durante el Congreso *se escucharon llamados* a que *se reconozcan los errores del pasado* y para que el partido <coref:de>*se adapte*</coref:de> al nuevo clima pol?tico.
(During the conference *calls were heard* that *past errors should be recognised* and that the party *should adapt itself* to the new political climate.)

### 8.1.5 Other phrases without a head noun

Proper names should be marked up wherever necessary (8.28), (8.29). They may be pre-modified, in which case the modifier should be included in the <coref:de>(8.30). However, where a proper name contains another NP within it, this smaller NP usually need not be separately marked up, since it typically works as a modifier and does not enter into anaphoric relations: in (8.31), for example, the word *Parades* would not be individually marked, nor would *Ulster* in (8.32).

(8.28)

*France* came from behind to beat *Croatia* 2-1 and reach their first World Cup final at *the Stade de France*. (BBC)

(8.29)

But home fans at *the Stade de France* endured an agonising final 20 minutes after *Laurent Blanc* was shown the red card following a tussle with *Slaven Bilic*. (BBC)

(8.30)

*Prolific Davor Suker* gave the keeper no chance for his fifth goal of the tournament. (BBC)

(8.31)

*The independent Parades Commission* banned the Protestant march from entering the nationalist Garvaghy Road on Sunday. (BBC)

(8.31)

While they have held a largely peaceful protest at the entrance of the Garvaghy Road, watched by *the Royal Ulster Constabulary* and the British army, loyalist violence has erupted throughout the province in protest.

Other examples of phrases classified as NPs without having a head noun are phrases with adjectives or quantifiers as heads, as in (8.32) and (8.33). Gerundive clauses also function in a similar way to NPs, and should be annotated where necessary.

(8.32)

I prefer *the largest.* (D)

(8.33)

A few people found their way to the destination but *a great many* did not understand the directions. (D)

(8.34)

They had been accused of *ignoring the environment*.

### 8.1.6 Conjoined NPs

Where two or more NPs are conjoined or disjoined, it may be necessary to mark up the larger NP as well as the constituent NPs, depending on whether it is referred to later in the dialogue. In (8.35), for example, the coordinated NP *John and Luise* serves as antecedent for the plural pronoun *They*.

(8.35)

*John and Louise* went out for a fancy meal for *her* graduation.
*They* had a huge argument and *he* walked out without paying.

(8.36)

```
<coref:de ID="DE_40">
  <coref:de ID="DE_41">
    John
  </coref:de>
  and
  <coref:de ID="DE_42">
    Louise
  </ coref:de>
```

```
</coref:de>
went out for a fancy meal for
<coref:de ID="DE_43">
  her
</coref:de>
graduation.
<coref:de ID="DE_44">
  They
</coref:de>
had a huge argument and
<coref:de ID="DE_45">
  he
</coref:de>
walked out without paying.
```

### 8.1.7 Linguistic contexts that do not introduce discourse entities

While the discussion of markables above defines the class of elements which can be marked up, there are cases where the linguistic or discourse context of an NP make it inappropriate to mark it as a <coref:de> element. Although we assume that in most cases markables will be automatically identified by means of search patterns formulated in terms of the MATE query language, it is possible that the annotator may want not to annotate these NPs as <coref:de>s. Examples of this are predicate nominals, where the NP cannot be considered to introduce a discourse entity. Some such contexts are discussed below.

*Predicate nominals*

Where the noun phrase can clearly be identified as being predicative, this should not be marked up, as this type of phrase does not introduce a discourse entity.

(8.51)

```
John is a policeman
(cf. John is tall.)
```

Indefinite NPs in copular position are typically predicative and need not be annotated. A more complex case is the one in which the NP in copular position is definite, as in the following example:

(8.52)

```
John is the President of the USA.
```

In the case of sentences like (8.52) one might argue either that we have an explicit equality and therefore both NPs should be marked as <coref:de>s, or that we only need to mark one (presumably the subject) since only one discourse entity is introduced by this sentence. Again, the decision depends in part on the task: if the system is used for information extraction it may be useful to annotate all anaphoric relations, whereas in other cases annotating these NPs may be considered a waste of time. A cautious policy is that unless an NP of this type is clearly predicative, it should be marked up as a discourse entity.

*Appositional phrases*

There are a number of different types of appositive constructions. Basic apposition consists of two noun phrases with identical reference, which need not be contiguous (8.53), (8.54). However, the appositional phrase may not be a simple NP (8.55), (8.56), may be introduced or followed by a marker such as *say* or *included* (8.56), (8.57), (8.58), and may even enter into a restrictive apposition, lacking the distinctive commas (8.59), (8.60).

(8.53)

News of the sudden death of *the imprisoned opposition leader, Chief Moshood Abiola*, has shaken Nigeria. (BBC)

(8.54)

*An unusual present awaited him, a book on ethics* (QG)

(8.55)

*The reason that he gave, that he didn't notice the other car,...* (QG)

(8.56)

We should send *one of the engines at Avon, say engine E1,* to Bath to pick up the tanker car (T)

(8.57)

*Many people, my sister included, ...* (QG)

(8.58)

*Many professions, such as the legal profession, ...* (GG)

(8.59)

*The famous critic Paul Jones* (QG)

(8.60)

*Your duty to report the accident* takes precedence over everything else (QG)

Our suggestion is to follow what proposed in MUCSS (Hirschman 1997), and tag the NP as a whole as well as any separate NP contained in the appositive clauses, if the appositive clause is

contiguous to the NP (8.61), (8.63). Discontinuous appositions can be marked separately (8.62). An ident link can then be marked between the appositive clause and either the other clause (discontinuous apposition) or the NP as a whole (see below). In the case of restrictive appositions, only the NP as a whole will be marked (8.64), (8.65).

(8.61)

```
News of the sudden death of
<coref:de ID="DE_44">
  the imprisoned opposition leader,
  <coref:de ID="DE_45">
    Chief Moshood Abiola
  </coref:de>
</coref:de>
, has shaken Nigeria
```

(8.62)

```
<coref:de ID="DE_46">
  An unusual present
</coref:de>
awaited him,
<coref:de ID="DE_47">
  a book on ethics
</coref:de>
```

(8.63)

```
We should send
<coref:de ID="DE_50">
  one of
  <coref:de ID="DE_51">
    the engines at Avon
  </coref:de>
  , say
  <coref:de ID="DE_52">
    engine E1
  </coref:de>
</coref:de>
, to Bath to pick up the tanker car
```

(8.64)

```
<coref:de ID="DE_53">
  the famous critic Paul Jones
</coref:de>
```

(8.65)

```
<coref:de ID="DE_54">
  Your duty to report the accident
</coref:de>
takes precedence over everything else
```

With appositions we have the same problem as with predicate nominals: some appositional phrases are really just predicative, and need not be marked up separately (see Predicate nominals).

(8.66)

```
Norman Jones, at that time a student,... (QG)
```

(8.67)

```
<coref:de ID="DE_48">
  Norman Jones, at that time a student.
</coref:de>..
```

(8.68)

```
<coref:de ID="DE_49">
  Julius Caesar, a well-known emperor
</coref:de>
```

whereas in the following case it may be argued that the appositional phrase does introduce a discourse entity which is equated with the one introduced by *Julius Caesar*:

(8.69)

```
Julius Caesar, the well-known emperor
```

Our only recommendation is to treat the two cases in the same way.

*Negated or questioned contexts*

Where the existence of an entity is being denied by the context in which the noun phrase occurs, as in (8.70), the discourse entity introduced by that entity often will not enter in any anaphoric relations; in these cases, the designer of a scheme may decide to mark the string as a <coref:de>only if it enters into subsequent anaphoric relations, as in (8.71).

(8.70)

```
But the volume of noise from the home fans began to subside
as the opening goal failed to materialise. (BBC)
```

(8.71)

```
I don't want to buy a car. It would cost me too much money.
```

A similar problem arises with NPs which occur as part of a question. While NPs within questions often don't really refer to an item in the world, annotators may wish to mark these items up in order to link them with another reference to an object; a link which may be made in the answer to the question (8.72). However, it should be noted that this is not really a co-specification relation;

see the section Instantiation in the extended scheme for a full treatment of <coref:de> in questions.

(8.72)

```
Does anyone have a pencil?
Yes - there's one there
```

*Disfluencies*

Disfluencies and repairs are common features of spoken dialogues. In this case, the problem is to decide whether noun phrases introduced within repaired parts of a dialogue should be marked up. As some theories suggest that repaired items are not available for subsequent processing, and to avoid unnecessary marking up of elements, it is suggested to make the marking optional: only those items within repaired sections which are subsequently referred to should be marked up. Note that if the repaired item corefers with an earlier <coref:de>and then is subsequently referred to, the later reference may be linked directly with the initial one, making the marking up of the repaired item unnecessary in this case. In (8.73), only the most complete phrase fragment needs to be marked up for the presence of *elles*, coreferential with *ces deux fusées*.

(8.73)

```
193 F: Donc qu'est ce qui / qu'est ce qui serait commun à ces deux fusées. Ces deux fusées ont /
194 D: c'est qu'elles ont / elles ont la même / elles / elles / toutes les / tous les ailerons
```

(In this case we deviate from the approach taken in DRAMA by Passonneau (1996), who proposes to mark up all NPs occurring in disfluencies, although again they do not create ambiguity.)

### 8.1.8 Discontinuous elements

Sometimes a discourse entity is not introduced by a single continuous phrase, but by a number of different utterances interrupted by disfluencies or comments. Sometimes this may be due to the way the text is segmented in the basic level: in the TRAINS corpus, for example, a great number of `<coref:de>` appear discontinuous due to the way in which the corpus has been marked up, with each speaker's utterance split into small strings over many numbered lines (8.45). If no comments or unrelated utterances intervene, it may be possible to simply group a number of lines into a single utterance, that can then be marked up as `<coref:de>`. If, however, the original separation into utterances has to be preserved, we propose to rely on the fact that information about discontinuous constituency is represented at the chunk level by means of next and prev attributes (see (8.46)), and to make the `<coref:de>` element point to the entire span of chunks beginning with the first chunk that is actually part of the NP (*one of the* in (8.45)) until the last chunk (*say engine E2* in the same example), as in (8.47).

(8.45)

```
9.6:  I think what we should do
9.7:  is
9.8:  hook up
```

```
9.9:  uh one of the
[2sec]
9.10: engines
9.11: uh
9.12: at Elmira
9.13: say engine E2
```

(8.46)

ch.xml

```
9.6:  I think what we should do
9.7:  is
9.8:  hook up
9.9:  uh <ch id="ch_60 ">one of the</ch>
[2sec]
9.10: <ch ID="ch 61" next="ch_63">engines</ch>
9.11: <ch ID="ch 62">uh</ch>
9.12: <ch ID="ch_63" prev="ch_61"> at Elmira </ch>
9.13: <ch ID="ch_64">say</ch>
      <ch ID="ch_65">engine E2</ch>
```

(8.47) coref.xml:

ch.xml

```
<coref:de ID="DE_01" href="ch.xml#id(ch_61)..ch.xml#id(ch_65)"/>
<coref:de ID="DE_02" href="ch.xml#id(ch_65)"/>
```

In (8.48), the giver's `<coref:de>` is interrupted by the follower's turn; again the mechanism for dealing with discontinuous constituents with chunks must be used to reconstruct it:

(8.48)

```
GIVER:    curving, just curving round the diamond
FOLLOWER: uh-huh
GIVER:    mine......uh-huh
```

(8.49)

ch.xml

```
GIVER:    Curving, just curving round
          <ch ID="ch_66" next="ch_68">
            the diamond
          </ch>
FOLLOWER: <ch ID="ch_67">
            uh-huh
          </ch>
GIVER:    <ch ID="ch_68" prev="ch_66">
            mine
          </ch> .
          <ch ID="ch_69">
            .....uh-huh
          </ch>
```

(8.50)

coref.xml

```
<coref:de ID="DE_01" href="ch.xml#id(ch_66)..id(ch_68)"/>
```

## 8.2 Assigning Links

This section discusses a few issues concerning the specification of links between discourse entities.

### 8.2.1 Coordinated NPs

Where two or more NPs are conjoined or disjoined, it may be necessary to mark up the larger NP as well as the constituent NPs, depending on whether it is referred to later in the dialogue. In (8.74), for example, the coordinated NP *John and Luise* serves as antecedent for the plural pronoun *They*.

(8.74)

```
John and Louise went out for a fancy meal for her graduation.
They had a huge argument and he walked out without paying.
```

(8.75)

```
<coref:de ID="de_40">
  <coref:de ID="de _41">
    John
  </coref:de>
  and
  <coref:de ID="de _42">
    Louise
  </coref:de>
</coref:de>
went out for a fancy meal for
<coref:de ID="de_43">
  her
</coref:de>
graduation.
<coref:de ID="de_44">
  They
</coref:de>
had a huge argument and
<coref:de ID="de_45">
  he
</coref:de>
walked out without paying.
<coref:link href="coref.xml#id(de_43)" type="ident">
  <coref:anchor href="coref.xml#id(de_42)"/>
</coref:link>
<coref:link href="coref.xml#id(de_44)" type="ident">
  <coref:anchor href="coref.xml#id(de_40)"/>
</coref:link>
<coref:link href="coref.xml#id(de_45)" type="ident">
  <coref:anchor href="coref.xml#id(de_41)"/>
</coref:link>
```

Note that in this example, discourse entity `de_40` is a set which includes both *John* and *Mary*; the relation between these entities cannot be annotated using only ident, but it's possible to do so with the extended set of relations discussed below.

### 8.2.2 Possessive pronouns

Possessive pronouns co-specify with their antecedents (e.g. *Louise ...her graduation*), and are therefore marked as ident, as shown in (8.74). The relationship between the whole NP designating the possessed item and its possessor, however, is not identity; *Louise* and *her graduation* clearly do not refer to the same entities in the world. Again, these relationships are part of the extended scheme: see Extended scheme: possessive.)

### 8.2.3 Clitics

As discussed in Section 7.2.2 above, we propose to use `<coref:seg>` elements to mark anaphoric expressions such as clitics which are morphologically incorporated into a verb. We then use `<coref:link>` elements to mark the anaphoric relations of the discourse entity referred to by a clitic with other discourse entities. So, for example, the anaphoric information in (8.38) would be annotated as follows:

(8.76)

```
A: Mira, te doy <coref:de id="de_1">este libro</coref:de>
   Â¿Conoces a <coref:de id="de_2">mi suegra?</coref:de>
   Pues <coref:seg id="seg_3">dáselo</coref:seg>
   cuando <coref:de id="de_5">la</coref:de>veas.

<coref:link href="coref.xml#id(seg_3)" type="ident">
  <coref:anchor href="coref.xml#id(de_2)"/>
</coref:link>
<coref:link href="coref.xml#id(de_5)" type="ident">
  <coref:anchor href="coref.xml#id(de_2)"/>
</coref:link>
```

### 8.2.4 Appositions

Depending on the application, it may be useful to mark an ident link between the appositive clause and either the other clause (discontinuous apposition) or the NP as a whole.

(8.77)

```
News of the sudden death of
<coref:de ID="de_44">
  the imprisoned opposition leader,
  <coref:de ID="de_45">
    Chief Moshood Abiola
  </coref:de>
</coref:de>
, has shaken Nigeria
<coref:link href="coref.xml#id(de_45)" type="ident">
  <coref:anchor href="coref.xml#id(de_44)"/>
</coref:link>
```

(8.78)

```
<coref:de ID="de_46">
  An unusual present
</coref:de>
awaited him,
<coref:de ID="de_47">
  a book on ethics
</coref:de>
<coref:link href="coref.xml#id(de_46)" type="ident">
  <coref:anchor href="coref.xml#id(de_47)" />
</coref:link>
```

(8.79)

```
We should send
<coref:de ID="de_50">
  one of
  <coref:de ID="de_51">
    the engines at Avon
  </coref:de>
  , say
  <coref:de ID="de_52">
    engine E1
  </coref:de>
</coref:de>
, to Bath to pick up the tanker car
<coref:link href="coref.xml#id(de_52)" type="ident">
  <coref:anchor href="coref.xml#id(de_50)"/>
</coref:link>
```

### 8.2.5 Coding Procedure for *<coref:link>* elements

The annotation should proceed in two steps: first all <coref:de>elements should be marked and agreed upon by the markers, then all links should be established. No convention on choosing a particular textual element as antecedent is needed, provided that the tool used supports coreference chain; and anyway they can be computed by hand.

## 8.3 Extending the set of anaphoric relations

In this section we discuss various issues that arise when trying to annotate more complex anaphoric relations than simple identity. As mentioned in Section 7, this can be done using the markup elements introduced in Section 4, but allowing more values for the type attribute of the `<coref:link>` element; we provide a specification of the modified `<coref:link>` element below. Our aim in this section is to highlight some of the problems that arise when doing so and suggest ways of reducing them. The set of relations allowed by the scheme derives from the analysis of Vieira (1998) and includes the bridging relations in DRAMA.

As the poor reliability scores which have been obtained by Poesio and Vieira (1998) for this kind of scheme indicate, once one moves beyond the **ident** relation, it can be difficult to decide how to classify the link between two elements. We addressed this problem by adopting the TEI technique of specifying 'subtypes' of links: in those cases in which it may be difficult to identify precisely the type of relation that exists between two entities, we introduced a more general relation to be used as **type** of a link, as well as more specific relations to be used as values of the subtype attribute in those cases in which this additional specification is possible.

### 8.3.1 Links with Extended Relations

***Description***

The <coref:link> element in the Extended Relations Scheme has two attributes: type and subtype. The type attribute is used to specify the semantic relation between the discourse entity introduced by a textual element and a previous discourse entity; the relations allowed include, in addition to identity, many of the relations often grouped under 'bridging' relations (Clark, 1977).

***Data Source***

The basic level for link relations is the same as discussed in Section 4.

***Segmentation***

As above, link elements do not mark parts of text.

***Assignment***

#### 8.3.1.1 Set Relations

*Member*

The **member** value should be used for the **type** attribute where the discourse entity pointed at by the coref:link element is a member of the set denoted by the discourse entity pointed at by the coref:anchor element. In the preferred reading of (8.82), for example, Paul and Jane are understood to be members of the set denoted by *the kids*. Note that this relation can apply whether it is the member or the set that appears first in the discourse, but when marking it up, the order of the arguments obviously matters.

(8.82)

```
The kids went to a party last weekend. Paul wanted to wear
his new suit, but Jane insisted on wearing her jeans.
```

(8.83)

```
<coref:de ID="de_85">
  The kids
</coref:de>
went to a party last weekend.
<coref:de ID="de_86">
  Paul
</coref:de>
wanted to wear his new suit, but
<coref:de ID="de_87">
  Jane
</coref:de>insisted on wearing her jeans
<coref:link href="coref.xml#id(de_86)" type="member">
  <coref:anchor href="coref.xml#id(de_85)"/>
</coref:link>
<coref:link href="coref.xml#id(de_87)" type="member">
  <coref:anchor href="coref.xml#id(de_85)"/>
</coref:link>
```

*Subset*

This value can be used when one discourse entity denotes a subset of the set denoted by the other discourse entity. As in the case of the element relation, the order of the arguments is important when marking up: the subset should be pointed at by the <coref:link> element, whereas the superset should be pointed at by the <coref:anchor> element. In the following example, there are two subsets of the initial set of rockets: the rockets which flew well, and the rockets which didn't fly well.

(8.84)

```
F: Alors donc / vous avez / ici / les modèles de fusées /
M: Oui
F: Et vous allez essayer de vous mettre d'accord sur un classement /
   hein classer les fusées qui ont bien volé ou qui ont moins bien volé /
```

(8.85)

```
F: Alors donc / vous avez / ici /
   <coref:de ID="de_88">les modèles de fusées</coref:de>
M: Oui
F: Et vous allez essayer de vous mettre d'accord sur un classement /hein classer
   <coref:de ID="de_89">les fusées qui ont bien volé</coref:de>
   ou
   <coref:de ID="de_90">qui ont moins bien volé</coref:de>
<coref:link href="coref.xml#id(de_89)" type="subset">
  <coref:anchor href="coref.xml#id(de_88)"/>
</coref:hlink
<coref:link href="coref.xml#id(de_90)" type="subset">
  <coref:anchor href="coref.xml#id(de_88)"/>
</coref:hlink
```

### 8.3.1.2 Possessive relations

Discourse entities can enter in a number of relationships that could be generically be described as cases of 'possession', and it's not always easy to decide precisely which type of relation is involved in any given case. In these cases, we propose to use the relation type **poss**; if a more detailed annotation is required, one of three subtypes can be specified - **attribute**, **partitive**, or **strict possession**.

*Attribute*

This relation is used when one <coref:de> expresses something which is an attribute of another <coref:de>; canonical examples of this include someone's height or weight. This relation may be expressed in two main ways: using a possessive pronoun or a genitive (*our sheer effort, his team's application*), or by means of an *of*-construction (*The quality of both teams, la taille de ailerons*). In both cases, the relation can be annotated by means of a <coref:anchor> element of type **poss**, subtype **attr**, between the whole NP and the NP denoting the possessor (8.86), (8.87), (8.88), (8.89). The order of the arguments is important: the <coref:link> element should point at the NP denoting the attribute, whereas the <coref:anchor> element should point at the NP denoting the

possessor. (If the possessor has been previously mentioned, then an **ident** link would also be marked between the two mentions of the possessor.) Note that in (8.87) two possessive relations are annotated: a strict possessive link to *Aime Jacquet* for *his team*, and an attributive link for *his team's application*.

(8.86)

```
French boss Aime Jacquet praised his team's application (BBC)
```

(8.87)

```
<coref:de ID="de_91">
  French boss Aime Jacquet
</coref:de>
praised
<coref:de ID="de_92">
  <coref:de ID="de_93">
    <coref:de ID="de_94">
      his
    </coref:de>
    team's
  </coref:de>
  application.
</coref:de>
<coref:link href="coref.xml#id(de_94)" type="ident">
  <coref:anchor href="coref.xml#id(de_91)"/>
</coref:link>
<coref:link href="coref.xml#id(de_93)" type="poss" subtype="sposs">
  <coref:anchor href="coref.xml#id(de_94)"/>
</coref:link>
<coref:link href="coref.xml#id(de_92)" type="poss " subtype="attr">
  <coref:anchor href="coref.xml#id(de_93)"/>
</coref:link>
```

(8.88)

```
He said: "I think our sheer effort and mental concentration saw us through."
```

(8.89)

```
He said: "I think
         <coref:de ID="de_95">
           <coref:de ID="de_96">
             our
           </coref:de>
           sheer effort and mental concentration
         </coref:de> saw us through."
<coref:link href="coref.xml#id(de_95)" type="poss " subtype="attr">
  <coref:anchor href="coref.xml#id(de_96)"/>
</coref:link>
```

(8.90)

```
F:  ...les ailerons...
M:  la taille de ailerons
F:  ...the wings...
M:  the height of the wings
```

(8.91)

```
F: ...
   <coref:de ID="de_97">
       les ailerons
      </coref:de>
   ...
M: <coref:de ID="de_98">
     la taille de
     <coref:de ID="de_99">
       ailerons
     </coref:de>
   </coref:de>
<coref:link href="coref.xml#id(de_98)" type="poss " subtype="attr">
  <coref:anchor href="coref.xml#id(de_99)"/>
</coref:link>
```

(8.92)

```
Team mate Rivaldo acknowledged the quality of both teams.
```

(8.93)

```
<coref:de ID="DE_100">
  Team mate Rivaldo
</coref:de>
acknowledged
<coref:de ID="DE_101">
  the quality of
  <coref:de ID="DE_102">
    both teams
  </coref:de>
</coref:de>.
<coref:link href="coref.xml#id(de_101)" type="poss" subtype="attr">
  <coref:anchor href="coref.xml#id(de_102)" />
</coref:link>
```

*Part*

A `<coref:anchor>` with type poss and subtype part is used where one `<coref:de>`denotes a physical part of another `<coref:de>`. Where the two objects are linked within one phrase, the link is marked in the same way as an attr link (8.95). Where the two `<coref:de>` are separately expressed, they each form the argument of the part link, with the part being the first argument, the whole the second. (Note that in order to annotate expressions like *the chair leg*, which have one sense very similar to that of possessive expressions like *the chair's leg*, it would be necessary to mark nominal premodifiers, contrary to what suggested in 8.1.1.)

(8.94)

```
The seat of the chair broke when I stood on it to open the window.
```

(8.95)

```
<coref:de ID="DE_104">
  The seat of
  <coref:de ID="DE_103">
    the chair
  </ coref:de>
```

```
</coref:de>
broke when I stood on
<coref:de ID="DE_105">
  it
</coref:de>
to open the window.
<coref:link href="coref.xml#id(de_101)" type="poss" subtype="part">
  <coref:anchor href="coref.xml#id(de_102)"/>
</coref:link>
```

(8.96)

```
F: donc est-ce que ces deux fusées ont les même ailerons? (MF)
So do these two rockets have the same wings?
```

(8.97)

```
donc est-ce que
<coref:de ID="de_105">
  ces deux fusées
</coref:de>
ont
<coref:de ID="de_106">
  les même ailerons
</coref:de>
<coref:link href="coref.xml#id(de_106)" type="poss" subtype="part">
  <coref:anchor href="coref.xml#id(de_105)"/>
</coref:link>
```

(8.98)

```
Army experts in Northern Ireland have defused a 1400 pound bomb left near
the main road in County Tyrone. The device, which included two booster
tubes, may have been designed for an attack on a security force patrol.
```

(8.99)

```
Army experts in Northern Ireland have defused
<coref:de ID="de_107">
  a 1400 pound bomb left near the main road in County Tyrone
</coref:de>
.
<coref:de ID="de_108">
  The device
</coref:de>
, which included
<coref:de ID="de_109">
  two booster tubes
</coref:de>
, may have been designed for an attack on a security force patrol
<coref:link href="coref.xml#id(de_108)" type="ident " >
  <coref:anchor href="coref.xml#id(de_107)"/>
</coref:link>
  <coref:link href="coref.xml#id(de_109)" type="poss" subtype="part">
<coref:anchor href="coref.xml#id(de_108)"/>
</coref:link>
```

*Strict possession*

A link of type poss and subtype sposs encodes the relationship between two objects where one 'belongs' to the other; typically, the possessor is a person or animate object. This link can be expressed, like the attributive construction, by a genitive or possessive pronoun (8.100), or by an *of*-construction (8.102). The order of the arguments is the same as in the attr link - possession first, possessor second.

(8.100)

```
It was a brave decision by Jerry Seinfeld to turn down $5m an episode
to make another series of his hugely popular sitcom. (BBC)
```

(8.101)

```
It was a brave decision by
<coref:de ID="de_110">
  Jerry Seinfeld
</coref:de>
to turn down $5m an episode to make another series of
<coref:de ID="de_111">
  <coref:de ID="de_112">
    his
  </coref:de>
  hugely popular sitcom
</coref:de>
<coref:link href="coref.xml#id(de_112)" type="ident">
  <coref:anchor href="coref.xml#id(de_110)"/>
</coref:link>
<coref:link href="coref.xml#id(de_111)" type="poss" subtype="sposs">
  <coref:anchor href="coref.xml#id(de_110)"/>
</coref:link>
```

(8.102)

```
The service is to be held in the Church of Our Lady
and St Patrick, in Ballymoney. (BBC)
```

(8.103)

```
The service is to be held in
<coref:de ID="de_113">
  the Church of
  <coref:de ID="de_114">
    Our Lady and St Patrick
  </coref:de>
</coref:de>
, in Ballymoney.
<coref:link href="coref.xml#id(de_113)" type="poss" subtype="sposs">
  <coref:anchor href="coref.xml#id(de_114)"/>
</coref:link>
```

To see how sometimes it can be difficult to distinguish between the three types of possessive links, consider in (8.104) - both *of*-constructions could be considered in a way to be strict possession, or the first could be an attribute and the second a part:

(8.104)

```
The health of Ronaldo, in the hours leading up to Sunday's World Cup final,
is dominating the sports pages of newspapers worldwide.
```

### 8.3.1.3 Other relations

In this section we illustrate a few other relations that may occur between two discourse entities. The designer of the annotator scheme may decide to annotate these or not depending on the degree of precision needed; else, a simple 'general relation' may be annotated.

*Bound anaphors*

This value should be used for type when a discourse entity is bound by a quantifier (8.105), (8.106). The pronoun and its antecedent are linked by a bound link, with the first argument of the link being bound by the second.

(8.105)

```
Nobody likes to lose his job.
```

(8.106)

```
<coref:de ID="de_80">
  Nobody
</coref:de>
likes to lose
<coref:de ID="de_81">
  his
</coref:de>
job
<coref:link href="coref.xml#id(de_81)" type="bound">
 <coref:anchor href="coref.xml#id(de_80)"/>
</coref:link>
```

(8.107)

```
Every man for himself.
```

*Function-value*

The f-v value can be used to indicate the relationship between a function and its value(s). Although these objects have the same reference when the function NP denotes a value, distinguishing this relation from identity is useful in those cases in which it is the sense of the NP that matters, as when a single function is assigned two different values (8.108) - marking these links as ident would result in asserting that 90 degrees is identical with 70 degrees. In this case, we would mark up two f-v links, one between the temperature and 90 degrees, and another between the temperature and 70 degrees (8.109). Because the f-v link is not symmetrical or transitive, unlike the ident link, this does not lead to 70 degrees and 90 degrees being marked as ident. The first argument of the link is the function, the second the value.

(8.108)

```
The temperature rose to 90 degrees before dropping to 70 degrees
```

(8.109)

```
<coref:de ID="DE_82">
  The temperature
</coref:de>
rose to
<coref:de ID="DE_83">
  90 degrees
</coref:de>
before dropping to
<coref:de ID="DE_84">
  70 degrees
</coref:de>
<coref:link href="coref.xml#id(de_82)" type="f-v">
  <coref:anchor href="coref.xml#id(de_83)"/>
</coref:link>
<coref:link href="coref.xml#id(de_82)" type="f-v">
  <coref:anchor href="coref.xml#id(de_84)"/>
</coref:link>
```

*Instantiation*

This relationship holds between two discourse entities when the second <coref:de> refers to a particular instantiation of the class identified by the first <coref:de>, as in (8.110). The link is marked up as type inst, with the first argument being the instance, and the second the class or non-referential use.

(8.110)

```
A: We need oranges.
B: There are some at Corning.
```

(8.111)

```
A: We need
   <coref:de ID="de_115">
     oranges
   </coref:de>.
B: There are
   <coref:de ID="de_116">
     some
   </coref:de>
   at Corning.
<coref:link href="coref.xml#id(de_116)" type="inst">
  <coref:anchor href="coref.xml#id(de_115)"/>
</coref:link>
```

This type of link might also be used to mark up the relationship between the class of entities whose existence or identity is queried by a question, and an entity that verifies that description:

(8.112)

```
A: which route do you want to take?
B: the Corning to Elmira route.
```

(8.113)

```
A: <coref:de ID="de_116a">
     which route
   </coref:de>
   do you want to take?
B: <coref:de ID="de_115a">
     the Corning to Elmira route
   </coref:de>.
<coref:link href="coref.xml#id(de_115a)" type="inst">
  <coref:anchor href="coref.xml#id(de_116a)"/>
</coref:link>
```

(8.114) illustrates the difficulty of marking up this kind of relation: the first mention of *un train* is not referential, and the last clearly is, but it is not so clear what the link should be between either of these and the mention in turn O4; this one is talking about the same hypothetical train as the first mention, but is still not referential, and therefore cannot be inst. We have therefore tentatively linked these first two non-referential uses as ident.

(8.114)

```
C2:-- est-c'que vous pourriez me dire si: il y a un train vers les douze heures quarante-cinq
      au départ de Paris Saint-Lazare pour Pontoise?
O3:-- pour Pontoise ?
C3:-- oui
O4:-- un train qui circule tous les jours ?
C4:-- oui
O5:-- ne quittez pas s'il vous plaît
.....
O6:-- allo
C5:-- oui
O7:-- (h) oui vous avez un train à douze heures quarante-cinq hein, il circule tous les jours
      sauf les dimanches et fêtes (SNCF)
C2:-- Could you tell me if there's a train around twelve forty-five from Paris Saint-Lazare to
      Pontoise?
O3:-- to Pontoise?
C3:-- Yes.
O4:-- A train which runs every day?
C4:-- Yes.
O5:-- Please hold the line.
......
O6:-- Hello?
C5:-- Yes.
O7:-- Yes, you've got a train at 12:45; it runs every day except Sundays and holidays.
```

(8.115)

```
C2:-- est-c'que vous pourriez me dire si: il y a
      <coref:de ID="de_117">
        un train
      </coref:de>
      vers les douze heures quarante-cinq au départ de Paris Saint-Lazare pour Pontoise?
O3:-- pour Pontoise ?
C3:-- oui
O4:-- <coref:de ID="de_118">
        un train qui circule tous les jours
      </coref:de>?
C4:-- oui
O5:-- ne quittez pas s'il vous plaît
....
O6:-- allo
C5:-- oui
O7:-- (h) oui vous avez
      <coref:de ID="de_119">
        un train
      </coref:de>
      à douze heures quarante-cinq hein,
      <coref:de ID="de_120">
        il
```

```
      </coref:de>
      circule tous les jours sauf les dimanches et fêtes
<coref:link href="coref.xml#id(de_117)" type="ident ">
  <coref:anchor href="coref.xml#id(de_118)"/>
</coref:link>
<coref:link href="coref.xml#id(de_119)" type="inst">
  <coref:anchor href="coref.xml#id(de_118)"/>
</coref:link>
<coref:link href="coref.xml#id(de_120)" type="inst">
  <coref:anchor href="coref.xml#id(de_119)"/>
</coref:link>
```

An alternative analysis of this example could be as follows: question C2 queries whether the set of trains for Pontoise from Paris Saint-Lazaire is non empty; O4 introduces a new class of trains, which however specializes the first class. So the link between de_120 and de_119 could be analyzed as either a subset relation or perhaps by introducing a new intensional relation between types, specializes. We will not discuss how to do this here.

*Event relations*

The event relation link encodes the link between a discourse entity and a preceding event or situation, expressed by a noun phrase, verbal phrase or sentence, in case the discourse entity plays a role of some sort in the event/situation. (This link is a generalization of the cause and arg links in DRAMA.) As in the case of possession relations, we propose a general link type e-rel; if further detail is required as to the role the <coref:de> plays in the event, e.g. to follow the DRAMA encoding, this may be provided in the subtype (e.g. cause, agent, patient, etc.); we will not attempt to define a general-purpose set of subtypes here. The coref:link element should point to the discourse entity, the coref:anchor to the event.

In the following example (from Passonneau), the e-rel relation holds between two noun phrases:

```
There was an explosion.  The noise was tremendous.
There was
<coref:de ID="de_3">
  an explosion.
</coref:de>
<coref:de ID="de_4">
  The noise
</coref:de>
was tremendous.
<coref:link href="id(de_4)" type="e-rel">
  <coref:anchor href="id(de_3)"/>
</coref:link>
```

In more complex cases, the event is introduced by a verb phrase or a sentence that has to be marked up. We propose to use the element <coref:seg (already used in the Core Scheme for annotating verbal elements containing clitics) for this purpose.

(8.116)

*Muslims from all over the world were taught gun-making and guerrilla warfare in Afghanistan. The instructors* were members of some of the most radical Islamic militant groups in the region. (Independent)

(8.117)

```
<coref:seg ID="de_130">
  Muslims from all over the world were taught gun-making and guerrilla warfare in Afghanistan.
</coref:seg>
<coref:de ID="de_131">
  The instructors
</coref:de>
were members of some of the most radical Islamic militant groups in the region.
<coref:link href="coref.xml#id(de_131)" type="e-rel">
  <coref:anchor href="coref.xml#id(de_130)"/>
</coref:link>
```

### 8.3.1.4 General

As mentioned above in the introduction to this section, the genrel (general relation) link may be used as a 'catch-all' label where an annotator believes that two discourse entities are related, but does not wish to give a very detailed classification of the type of non-ident relationship involved. Alternatively, this type of relation may also be used in addition to these classes to cover any other types of links which do not appear to fit into the above classification. One example of this can be seen in the phrase *The man who gives his paycheck to his wife is wiser than the man who gives **it** to his mistress*, in that 'it' here does not refer to the same entity as its antecedent, 'his paycheck', but rather to something which stands in the same relationship to the second man as the paycheck does to the first.

#### *Example*

Plenty of examples are given above.

#### *Coding Procedure*

As in the case of the MUCCS scheme, annotation with <coref:link> elements should follow the determination of <coref:de>elements. The decision concerning the value of **type** should be done as follows:

```
a. see if the current discourse entity is identical with a previous discourse entity;
   if so, create a link, and specify type=ident;
   b. else, see if it stands in one of the set relations;
   c. else, see if it stands in a possession relation;
   d. else, if it appears that the discourse entity is in a relation with one of the
      previous discourse entities,
      but the relation is not one of those listed above, create a link and then
          1. if additional attribute values are used, examine if one of those applies;
          2. else, use type=genrel
```

#### *Markup Table*

| <coref:link> | |
|---|---|
| id | [ASCII] |
| type | ident, member, subset, poss, e- |

| | |
|---|---|
| | rel, argptv, prop, bound, f-v, inst, genrel |
| subtype | attr, part, sposs, cause |
| href | <coref:de> |
| content | <coref:anchor> |

| <coref:anchor> | |
|---|---|
| id | [ASCII] |
| href | <coref:de> |

The mapping between the DRAMA relations and those discussed below is specified as follows:

| **DRAMA's name** | **Meta-scheme name** |
|---|---|
| Part | poss, subtype part |
| Cause | e-rel, subtype cause |
| Poss | poss, subtype sposs |
| arg (as part of arg/ptv) | e-rel |
| prop | not included |
| ptv (as part of arg/ptv) | e-rel |
| Coref | ident |
| Subset | subset-of |
| Member | member |

## 9 References

A. Anderson, M. Bader, E. Bard, E. Boyle, G. Doherty, S. Garrod, S. Isard, J. Kowtko, J. McAllister, J. Miller, C. Sotillo, H. Thompson, and R. Weinert (1991). The HCRC MapTask Corpus. *Language and Speech*, 34(4): 351-366.

Bruneseaux, F. and Romary, L. (1997) REG: Reference Encoding Guidelines, Draft, March 25, 1997.

Carletta, J. (1996) Assessing agreement on classification tasks: the kappa statistic, *Computational Linguistics* 22(2): 249-254.

Carlson, G. N. (1977). *Reference to Kinds in English*. PhD thesis, University of Massachusetts at Amherst

Clark, H. H. (1977) Bridging. In P. N. Johnson-Laird and P.C. Wason, editors, *Thinking: Readings in Cognitive Science*. Cambridge University Press, London and New York.

Fraurud, K. (1990) Definiteness and the processing of NPs in natural discourse.*Journal of Semantics*, 7:395-433.

D. Gross, J. Allen and D. Traum, The TRAINS 91 Dialogues, TRAINS Technical Note 92-1, 1993.

Grosz, B. J. (1977). *The Representation and Use of Focus in Dialogue Understanding*. PhD thesis, Stanford University.

Hawkins, J. A. (1978). *Definiteness and Indefiniteness*. Croom Helm, London.

Hirschman, L. (1997) MUC-7 Coreference Task Definition, Version 3.0. In *Proc. MUC-7* Karttunen, L. (1976). Discourse Referents. In J. McCawley, editor, *Syntax and Semantics 7 - Notes from the Linguistic Underground*. Academic Press, New York.

Partee, B. H. (1972) Opacity, coreference, and pronouns. In D. Davidson and G. Harman, editors, *Semantics for Natural Language*. D. Reidel, Dordrecht, Holland, pages 415-441.

Passonneau, R. J. (1996) Instructions for Applying Discourse Reference Annotation for Multiple Applications (DRAMA)

Poesio, M. and R. Vieira. (1998) A corpus-based investigation of definite description use. *Computational Linguistics*, n. 24, n.2.

Prince, E. F. (1981) Toward a taxonomy of given-new information. In P. Cole, editor, *Radical Pragmatics*. Academic Press, New York, pages 223-256.

Quirk and S. Greenbaum. 1973. *A University Grammar of English*, Longman.

Sidner, C. L. (1979) Towards a computational theory of definite anaphora comprehension in English discourse. Ph.D. thesis, MIT.

Vieira, R. (1998) Definite Description Resolution in Unrestricted Texts. Ph.D. thesis, University of Edinburgh, Centre for Cognitive Science.

Webber, B. (1978) A Formal Approach to Discourse Anaphora. Ph.D. thesis, Harvard University.

# Communication Problems

Laila Dybkjær and Amanda Schiffrin

## 1. Introduction - Coding Purpose

Co-operativity is a central issue both in human-computer dialogue and in human-human dialogue. Non-co-operative behaviour easily leads to miscommunication and an unnecessarily long, complicated and perhaps even ultimately unsuccessful dialogue. In particular in human-computer interaction the consequences may often be a totally failed dialogue because of the system's limited abilities to detect, handle and recover from non-co-operative (or seemingly non-co-operative) dialogue [Bernsen et al. 1996].

Research into aspects of communication, including various communication problems and co-operativity, is not new (see e.g. [Grice 1975]). However, there is no exhaustive theory on these, and much of the research has focused on human-human dialogue.

With an increasing number of advanced spoken language dialogue systems available as viable means of supporting people in carrying out ordinary tasks (such as automatic flight/train timetable information, ticket booking, directory enquiries, etc.), there is also an increasing demand for rigorous methods and tools for the analysis of problems (latent and actual) in communications so as to pre-empt them if possible, or failing this, to initiate some form of repair. Thus, the study of communication problems is of obvious importance for the development of adequate system interaction models.

If detected, communication problems in everyday conversation typically lead to clarification or repair meta-communication. Human-human dialogue both allows for and is greatly assisted by the possibility of this kind of repair and maintenance. However, for spoken language dialogue systems the situation is different. The possibility of real-time handling of clarification and repair meta-communication using current technology is seriously limited. User needs for clarification meta-communication that arise from the way the system addresses the domain, can easily surpass the systems meta-communication skills. Thus co-operative communication is important because it facilitates smooth interaction and prevents unnecessary user-initiated clarification and repair meta-communication, as well as other kinds of unexpected user behaviour with which the system cannot cope.

In spite of this, the detection of communication problems in spoken language dialogue systems has so far usually been carried out in a rather unsystematic manner, and on ad hoc basis; it has generally only been performed at a fairly late stage, as part of the evaluation, if at all. In order to support a more cost-efficient development process for interaction models of spoken dialogue systems we need a solid understanding of communication problems, both their nature and why they occur. A straightforward way of achieving this is to annotate them for analysis from the transcripts of SLDS interaction. Annotation of communication problems in spoken dialogue corpora may not only help developers and researchers to extract information on the deficiencies of a system, but may also yield clues as to how it might be improved. It could also provide the insight required for creating methods or tools to enable the efficient, systematic development and evaluation of a system, especially during early analysis and design. This would result not only in the improvement of interaction model quality, but also in the reduction of development costs.

Communication problems are in several respects different from most other phenomena usually annotated and studied in a corpus. Most notably they need not necessarily be present in a corpus at all. In fact the fewer there are of them the better. This is in direct contrast with prosodic and morpho-syntactic phenomena and dialogue acts, which are present in any spoken dialogue corpus. The same is also true to some extent for coreference (although unlike the aspects of language listed previously, mark-up for coreference is neither continuous nor contiguous). All these phenomena form part of the building blocks of a dialogue. Communication problems, on the other hand, are disruptive to a dialogue and co-operative human interlocutors usually try to avoid them. However, in order to avoid them in human-computer interaction we need to study the nature and mechanisms of communication problems. The computer is much less flexible than a human which makes it much more likely that communication problems will occur in a dialogue unless great care is taken to design the dialogue co-operatively within the limitations dictated by the spoken language dialogue system in question.

# 2. Existing Schemes

Annotation of communication problems is in its infancy and coding schemes tailored to the description of communication problems are very few. However, aspects of communication problems are often reflected in coding schemes for other levels, e.g. dialogue acts. Three of the four communication problems coding schemes presented in D1.1 [Klein et al. 1998] do not have a direct focus on communication problems. Rather, they include phenomena that relate to the level in focus, e.g. coreference or dialogue acts, as well as to communication problems. In fact many of the schemes in D1.1 do include one or more of these phenomena, which is a consequence of the cross-level nature of communication problems. Some communication problems are caused by flawed grammar or vocabulary design (i.e., errors at the morpho-syntactic level). Other problems may be due to misinterpretation or non-interpretation of coreference, and so on.

Among the existing schemes analysed only the Odense coding scheme is exclusively devoted to communication problems per se. It has therefore been natural to take this scheme as the point of departure for the communication problems level.

# 3. Selected Scheme (The Odense Scheme)

The Odense coding scheme was developed in support of the design of co-operative system utterances in spoken human-machine dialogue. The co-operativity guidelines [Bernsen et al. 1998, Dybkjær 1999], cf. Figure 1, which form the basis of the scheme were developed from a set of simulated human-machine dialogues. These guidelines were then compared to Grice's maxims [Grice 1975] and shown to significantly extend them. Finally its successful application to new Danish, English and German spoken human-machine corpora tested the coding scheme. Thus the guidelines are proved to be both theoretically founded and application-oriented.

## 3.1 Guidelines for Co-operative Communication

| Interaction Aspect | G/S G no. | Abbreviation | Generic or Specific Guideline |
|---|---|---|---|

| | | | |
|---|---|---|---|
| Aspect 1: Informativeness | GG1 | Say enough. | *Make your contribution as informative as is required (for the current purposes of the exchange). |
| | SG1 | State commitments explicitly. | Be fully explicit in communicating to users the commitments they have made. |
| | SG2 | Provide immediate feedback. | Provide feedback on each piece of information provided by the user. |
| | GG2 | Don't say too much. | *Do not make your contribution more informative than is required. |
| Aspect 2: Truth and evidence | GG3 | Don't lie. | *Do not say what you believe to be false. |
| | GG4 | Check what you say. | *Do not say that for which you lack adequate evidence. |
| Aspect 3: Relevance | GG5 | Be relevant. | *Be relevant, i.e. be appropriate to the immediate needs at each stage of the transaction. |
| Aspect 4: Manner | GG6 | Avoid obscurity. | *Avoid obscurity of expression. |
| | GG7 | Avoid ambiguity. | *Avoid ambiguity. |
| | SG3 | Ensure uniformity. | Provide same formulation of the same question (or address) to users everywhere in the system's interaction turns. |
| | GG8 | Be brief. | *Be brief (avoid unnecessary prolixity). |
| | GG9 | Be orderly. | *Be orderly. |
| Aspect 5: Partner asymmetry | GG10 | Highlight asymmetries. | Inform the users of important non-normal characteristics which they should take into account in order to behave co-operatively in spoken interaction. Ensure the feasibility of what is required of them. |
| | SG4 | State your capabilities. | Provide clear and comprehensible communication of what the system can and cannot do. |
| | SG5 | State how to interact. | Provide clear and sufficient instructions to users on how to interact with the system. |
| Aspect 6: Background knowledge | GG11 | Be aware of user's background knowledge. | Take partners' relevant background knowledge into account. |
| | SG6 | Be aware of user inferences. | Take into account possible (and possibly erroneous) user inferences by analogy from related task domains. |
| | SG7 | Adapt to novices and experts. | Separate whenever possible between the needs of novice and expert users (user-adaptive interaction). |
| | GG12 | Be aware of user expectations. | Take into account legitimate partner expectations as to your own background knowledge. |
| | SG8 | Cover the domain. | Provide sufficient task domain knowledge and inference. |
| Aspect 7: Repair and clarification | GG13 | Enable meta-communication. | Enable repair or clarification meta-communication in case of communication failure. |
| | SG9 | Enable system repair. | Initiate repair meta-communication if system understanding has failed. |
| | SG10 | Enable inconsistency clarification. | Initiate clarification meta-communication in case of inconsistent user input. |
| | SG11 | Enable ambiguity clarification. | Initiate clarification meta-communication in case of ambiguous user input. |

**Figure 1.** Guidelines for co-operative system interaction. Generic guidelines (indicated by the use of 'GG' at the start of a guideline name) are at the general level of Gricean maxims (marked by an asterisk * above), and are grouped into different aspects of interaction. Each specific guideline (indicated by 'SG') is subsumed by one of these generic guidelines.

The guidelines cover seven different aspects of interaction as shown in Figure 1. The distinction between guideline and aspect is an important one because an aspect serves to highlight the property of interaction addressed by a particular guideline, thus identifying dimensions of co-operativity over and above the level of the co-operative guidelines themselves.

There are two types of guidelines: generic (GG) and specific (SG). The specific guidelines are a refinement of the generic, and are thus subsumed by them. Generic guidelines are more general and express what to do or take into account when communicating. Specific guidelines specialise the generic guideline by which they are subsumed to certain classes of phenomena, explain how to do something expressed by the generic guideline, and are specifically aimed at system design. Although subsumed by generic guidelines, the specific guidelines are important in interaction design because they serve to elaborate on the kind of interaction model that the developer should be looking for when designing co-operative system behaviour.

It should be noted that not every generic guideline subsumes some specific guideline(s), and the specific guidelines do not add up to, or replace, the generic guideline from which they are derived. A given communication problem should always be described if possible by referring to the violation of a specific guideline if there is one which fits. Otherwise the reference should be to a generic guideline.

Another point of interest is that guidelines may at times support one another, but at other times conflict when applied during actual interaction design. When guidelines conflict, the designers have to trade off different design options against one another, perhaps for example by giving the options a weighting of some kind depending upon the guideline(s) referred to. When designing a system introduction, for instance, developers may find that GG2 (don't say too much) conflicts with GG1 (say enough), SG4 (tell what the system can and cannot do) and SG5 (instruct on how to interact with the system). If the introduction is long and complex, even if all the points made are valid and important, users tend to get bored and inattentive. On the other hand, if the introduction is brief or even non-existent, important information may have been left out, increasing the likelihood of miscommunication during task performance.

## 3.2 Scope of the Odense scheme

The Odense annotation scheme does not pretend to account for all possible communication problems. It was created with the purpose of improving spoken language dialogue system design and so far it has only been tested on dialogues which were

- shared goal,
- human-machine, and
- two-participant dialogue.

Thus the scheme is not claimed to be valid for human-human dialogues and non-shared goal dialogues since it has not been tested for this purpose.

*Shared-goal* dialogues are dialogues in which the interlocutors collaborate to achieve a common goal, such as booking a ferry ticket or getting/providing information about flight arrivals. Today's spoken language dialogue systems are shared-goal systems taking for granted that the user's goal is to carry out (one of) the task(s) that the system can perform. Shared-goal dialogue

should be seen in contrast to general conversation which is subject to possible conflicting goals and intentions between the participants.

Human-human dialogue has many more facets than today's spoken *human-machine* dialogue is capable of handling. Therefore, we cannot discount the possibility that there exist communication problems in human-human dialogue which would call for extra guidelines for co-operative dialogue behaviour not included among those in Figure 1. Human-human communication problems may for example derive from conflicting goals/intentions, gestural misunderstandings, talking 'above one's head', lying, and hidden agendas. Whether these potential problems in understanding between humans in dialogue can be captured by the annotation system developed and described here is still open to question. However, extending the current set of guidelines will be easy, using the coding module presented in Section 4.1.

So far the scheme has only been tested on *two-participant dialogues*. Spoken human-computer dialogue is normally between one human and one machine. It is likely that the communication problems will also apply to multi-party dialogues but this remains to be tested.

The primary focus of studies so far has been to mark up communication problems caused by the system because the emphasis was on investigating how system interaction could be improved to achieve a smoother dialogue with users. Of course users also commit errors from time to time which can be the direct cause of a communication problem. User errors have only been investigated to a limited extent in this connection (i.e., in their own right) [Bernsen et al. 1998] and we still lack detailed knowledge of their mechanisms. We are mainly interested in those cases of user errors that are triggered by inappropriate system interaction. However, it seems likely that the human interlocutor is able to cause the same categories of communication problems as the system does, i.e. by violating the guidelines listed in Figure 1.

## 3.3 Using the Odense scheme

The set of tags (elements and attributes) used by the Odense scheme is small and simple, even if a three-component structure is involved, cf. the bottom three coding files shown in Figure 2a. However, it is a non-trivial task to identify communication problems and analyse them correctly to determine which guidelines they violate and how they do it, i.e. what types of violation we are dealing with.

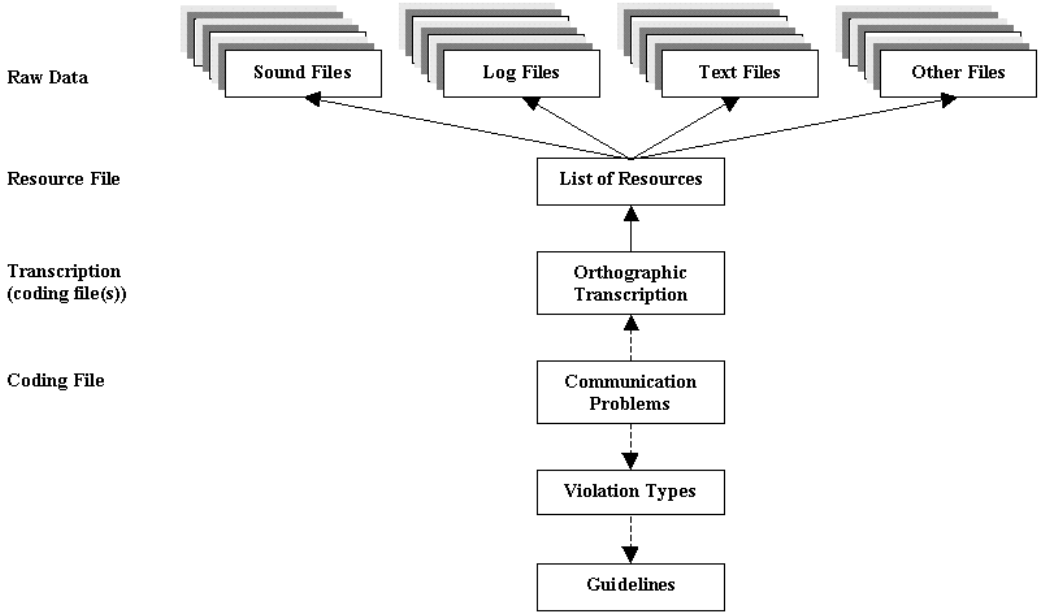



**Figure 2a.** File organisation for a corpus annotated with respect to communication problems. An arrow A B means that elements in A refer to elements in B by their attribute IDs, while an arrow A B means that there is a reference in A to B by its file name.

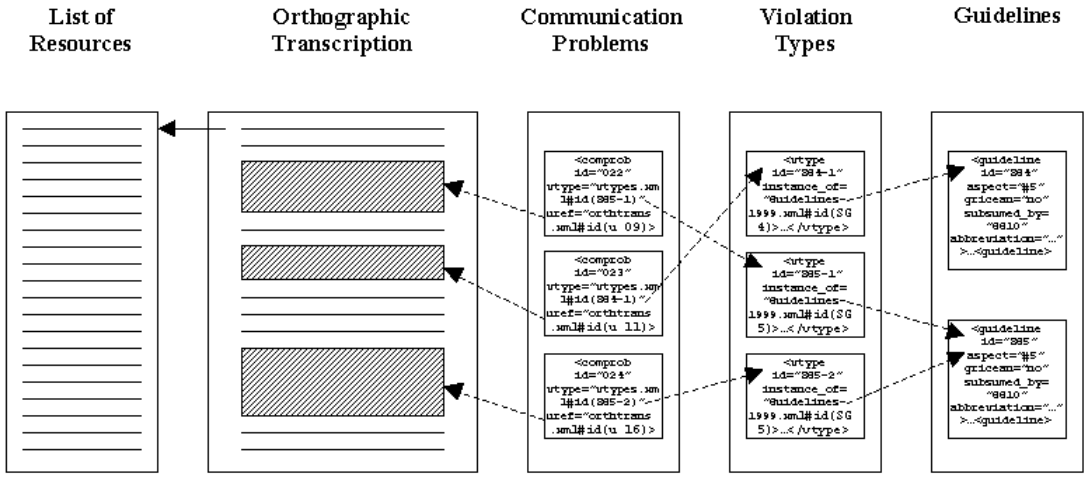



**Figure 2b.** Section of Figure 2a in detail. Please note that the orthographic transcription may refer to more than one resource in the list of resources.

During the detection and analysis of communication problems, an orthographic transcription of the dialogue is used; often the logfile will have to be inspected as well (cf. the reference structure in Figure 2a and cf. [Dybkjær et al. 1998]). In a few situations it may even be necessary to have access to the sound files or to a phonetic transcription in order to determine an ambiguous utterance in the orthographic transcription (which would be clarified in the spoken version due to intonation). For example, some questions have the same form as statements, and only the information provided by the intonation will reveal whether it is one or the other.

Communication problems are marked up as types of violation of the set of guidelines for co-operative communication in Figure 1 (both the generic guidelines and those specific to spoken

language dialogue systems). A guideline may be violated in several different ways; for example, GG7 (avoid ambiguity) would be violated by not saying whether a time "9 o'clock" given to the user by the system means 9 am or 9 pm. Another violation of the same guideline might be not to make clear whether a certain flight arrival time refers to that given by the timetable or to the actual expected arrival time.

Such violation types are necessarily task dependent as they refer to concrete problems found in dialogues within a particular system. Thus a communication problem refers to the part of the orthographic transcription in which the guideline violation was found and it also refers to a set of violation types (which is created along with the markup of communication problems). Each type of violation in its turn refers to the particular guideline that was violated. The reference structure for communication problems markup is therefore basically the one shown in Figure 2b.

To learn more about how to detect and analyse communication problems, using the guidelines in Figure 1 as a frame of reference, it is recommended that the reader take a look at [Dybkjær 1999] and [Bernsen et al. 1998]. These references contain a collection of examples of communication problems, violation types and references to the 24 guidelines listed in Figure 1. See also the examples in Figures 3-6 which show how communication problems are annotated and a types file established. Figures 3 and 4 show orthographic transcription of two Sundial dialogues (with sketchy annotation only). Figure 5 shows the guideline violation types file for the two dialogues and Figure 6 shows the corresponding communication problems file.

# 4. Markup Declaration

The structure for coding communication problems (shown in Figure 2b above) is reflected in the following three sections on guidelines, guideline violation types and communication problems. For each of these sections the structure used reflects the coding module description in [Dybkjær et al. 1998]. A coding module prescribes what constitutes a coding, including the representation of markup, and the relations to other codings.

Please note that elements are always automatically assigned unique identifiers. Thus there is no indication of identifiers in the attribute lists under markup declaration in the coding modules. Please also note that the examples in the coding modules illustrate what the contents may look like, i.e. examples of aspects, guidelines, violation types and communication problems. The markup is only inserted to show examples of use of the tags. The internal XML representation will certainly be different from the markup used in the examples. However, the user is not supposed to be concerned with the actual XML representation, but only with which tags are available and how they should be used. Readers who are interested XML and how the internal representation is generated behind the user interface are referred to the description in MATE deliverable D3.2.

## 4.1 Guideline Coding Module

**Name:** Guidelines.

**Coding purpose:** Records the different generic and specific guidelines, the violation of which typically leads to communication problems in a dialogue.

**Coding level:** Communication problems.

**Data sources:** List of generic and specific guidelines for co-operative dialogue design.

**Module references:** None.

**Markup declaration:**

```
ELEMENT aspect

ELEMENT guideline

ATTRIBUTES
  aspect: REFERENCE(this, aspect)

  gricean: ENUM (yes|no)

  subsumed_by: REFERENCE(this, guideline)

  abbreviation: TEXT
```

**Description:** Two elements are used to annotate the guidelines. One is `aspect`. `aspect` is used to indicate a grouping of the guidelines. For example, the 24 guidelines in Figure 1 are divided into seven groups or aspects. The element `aspect` has no explicit attributes. The mandatory attribute `id` which is a unique identifier, is always generated automatically for all elements.

A second element is `guideline` which marks up a particular guideline. `guideline` has four attributes.

`aspect` is mandatory. It is a reference to the aspect to which the guideline belongs. The `aspect` indicated for a specific guideline must always equal the `aspect` indicated for the generic guideline by which it is subsumed.

`gricean` is mandatory for guidelines which are the same as Grice's maxims [Grice 1975]. The `yes` value is used to indicate a maxim. For non-maxims `gricean` is optional. If indicated, the `no` value must be chosen. Using the value `yes` indicates whether a certain guideline is one of Grice's maxims.

`subsumed_by` should always be used for specific guidelines to indicate by which generic guideline it is subsumed. `subsumed_by` cannot be used for generic guidelines.

`abbreviation` is optional but recommended. It provides an abbreviated form of the guideline. It carries the essential meaning and may be easier to remember than the "canonical" expression of the guideline.

**Examples:**

```
<aspect id="1">Informativeness</aspect>
...
<aspect id="5">Partner asymmetry</aspect>
<guideline id="GG1" aspect="#1" gricean="yes" abbreviation="Say enough">
  Make your contribution as informative as is required (for the current purposes of the exchange).
</guideline>
<guideline id="SG1" aspect="#1" subsumed_by="#GG1" abbreviation="State commitments explicitly">
  Be fully explicit in communicating to users the commitments they have made.
</guideline>
<guideline id="SG2" aspect="#1" subsumed_by="#GG1" abbreviation="Provide immediate feedback">
  Provide feedback on each piece of information provided by the user.
</guideline>
<guideline id="GG2" aspect="#1" gricean="yes" abbreviation="Don't say too much">
  Do not make your contribution more informative then is required.
</guideline>
...
<guideline id="GG10" aspect="#5" abbreviation="Highlight asymmetries">
  Inform the users of important non-normal  bara teristi s whi h they should take into a  ount
```

```
  in order to behave co-operatively in spoken interaction.  Ensure the feasibility of what is
  required of them.
</guideline>
<guideline id="SG4" aspect="#5" subsumed_by="#GG10" abbreviation="State your capabilities">
  Provide clear and comprehensible communication of what the system can and cannot do.
</guideline>
...
```

**Coding procedure:**

The guidelines for co-operative dialogue design are part of the coding module for communication problems defined below. However, they may also be reused in other coding modules for communication problems. If a user defining a new communication problems module should want to build on a different set of guidelines it may well be that s/he can still reuse the coding module for guidelines defined here. Encoding a set of guidelines using the present coding module is not very complicated and the following procedure is recommended as sufficient:

1. Encode by coder 1.
2. Check by coder 2.

**Creation notes:**
Authors: Hans Dybkjær and Laila Dybkjær.
Version: 1 (25 November 1998), 2 (19 June 1999).
Comments: None.
Literature: [Bernsen et al. 1998, Dybkjær 1999].

## 4.2 Violation Types Coding Module

**Name:** Violation_types.

**Coding purpose:** Records the different ways in which generic and specific guidelines are violated in a given corpus, i.e. types of problems found in the corpus. The corpus is implicitly given by a communication problems coding file referring to the problem type coding file as well as to a transcription.

**Coding level:** Communication problems.

**Data sources:** List of types of violations of generic and specific guidelines for co-operative dialogue design. The list is generated during analysis of a corpus with respect to communication problems.

**Module references:** Module Guidelines.

**Markup declaration:**

```
ELEMENT vtype

ATTRIBUTES

      instance_of: REFERENCE(Guidelines, guideline)

      alternative_instances: REFERENCE(Guidelines, guideline+)
```

**Description:** Each description of a violation type is annotated by the element `vtype`. This element has two attributes.

The attribute `instance_of` is mandatory. `instance_of` is a reference to a particular guideline in a file which contains the guidelines for co-operative dialogue.

`alternative_instances` is optional. Guidelines overlap and in some cases the coder may be in doubt whether one or the other guideline was violated. The attribute `alternative_instances` allows the coder to express this doubt by letting him/her indicate one or more (this is what '+' means) other guidelines than the one referred to by `instance_of`.

The body of `vtype` contains the description of the actual type of violation.

**Example:**

```
<vtype id="SG4-1" instance_of="Guidelines-1999#SG4">
  Too little said on what system can and cannot do: BA often missing;
  time-table enquiries always missing.
</vtype>
```

**Coding procedure:** Each communication problem is seen as a certain type of violation of a guideline. The violation types are highly task dependent. The file containing these types is built in parallel with the analysis and markup of communication problems. This file is very special in the sense that its contents, i.e. the text, as well as the markup are created at the same time and by the coder. The contents are textual descriptions of the violation types. We recommend to use the same coding procedure for violation types as for markup of communication problems since the two actions are tightly connected. As a minimum the following procedure should be followed:

1. Encode by coders 1 and 2.
2. Check and merge codings (performed by coders 1 and 2 until consensus).

**Creation notes:**
Authors: Hans Dybkjær and Laila Dybkjær.
Version: 1 (25 November 1998), 2 (19 June 1999).
Comments: None.
Literature: [Bernsen et al. 1998, Dybkjær 1999].

## 4.3 Communication Problems Coding Module

**Name:** Communication_problems.

**Coding purpose:** Records the different ways in which generic and specific guidelines are violated in a given corpus. The communication problems coding file refers to a problem type coding file as well as to a transcription.

**Coding level:** Communication problems.

**Data sources:** Dialogue corpora.

**Module references:** Module Basic_orthographic_transcription; Module Violation_types.

**Markup declaration:**

```
ELEMENT comprob

ATTRIBUTES

      vtype:      REFERENCE(Violation_types, vtype)

      wref:       REFERENCE(Basic_orthographic_transcription, (w,w)+)

      uref:       REFERENCE(Basic_orthographic_transcription, u+)

      caused_by:  REFERENCE(this, comprob)

      temp:       TEXT

ELEMENT note

ATTRIBUTES

      wref: REFERENCE(Basic_orthographic_transcription, (w,w)+)

      uref: REFERENCE(Basic_orthographic_transcription, u+)
```

**Description:** In order to annotate communication problems caused by inadequate systems design we use the element `comprob`. It refers to some kind of violation of one of the guidelines listed in Figure 1. The `comprob` element may be used to mark up any part of the dialogue which caused the communication problem. Thus it may be used to annotate one or more words, an entire utterance or even several utterances in which a communication problem was detected. The `comprob` element has five attributes.

The attribute `vtype` is mandatory. `vtype` is a reference to a particular description of a guideline violation in a file which contains the different kinds of violations of the individual guidelines.

Either `wref` or `uref` must be indicated. Both these attributes refer to an orthographic transcription. `wref` delimits the word(s) which caused a communication problem, and `uref` refers to one or more entire utterances which caused a problem.

The attribute `caused_by` is optional. In some cases a communication problem in a dialogue will be caused by a problem which occurred earlier in that dialogue. `caused_by` is used to refer to a communication problem which was found elsewhere in the dialogue and which led to the present communication problem.

`temp` is an optional attribute. It indicates a temporary markup. It usually takes a few dialogues before the coder gets a good grasp of the types of guideline violations which tend to occur in the corpus and what caused them. Often logfile inspection will be needed to make an exact diagnosis. Moreover, some problems become easier to detect when comparing a few dialogues. Thus `temp` is mainly for use during initial markup of a corpus but may also be used later if it is practical to make some temporary notes before making the final diagnosis. The `vtype` attribute overrides whatever communication problems the attribute `temp` indicates.

In the beginning of the analysis the `vtype` attribute may be left open and the `temp` attribute filled in to describe the kind of guideline violation identified. Very soon, however, a file containing the violation types should be established and in most cases the `temp` comments can simply be moved to this file and possibly modified to provide a violation type description. Note that due to this and to the coding procedure requiring at least two coders the violation type references in the `vtype` attribute are likely to eventually be re-classified.

The note element can be used anywhere in a corpus to comment on whatever the user wants. It refers to one or more words or one or more utterances in the same way as the comprob element. The body of the note element contains text.

**Example:**
The following example communication problems markup assumes this snippet of a transcription from the Sundial corpus and refers to the example in the violation types coding module:

```
<u id="S1:7-1-sun" who="S">
  flight information british airways good day can I help you
</u>
<comprob id="3" vtype="Sundial_problems#SG4-1" uref="Sundial#S1:7-1-sun"/>
<note id="2" uref="Sundial#S1:7-1-sun">
  The system provides too little information about its capabilities and limitations.
  It is of course an ideal that little information is necessary.
  However, the risk is that the user will be misled and assume stronger or weaker system
  capabilities than are actually present.  Designers should look out for symptoms to this effect.
  The present introduction suggests that users can ask about anything to do with British Airways
  flights.
  No current system is likely to be able to do that.
  Another interpretation of the system's introduction is that it is owned by British Airways but
  can answer any question about flights.  The former interpretation seems the most natural one.
  So the system's opening probably should not be deemed ambiguous.
</note>
```

**Coding procedure:** We recommend to use the same coding procedure for markup of communication problems as for violation types since the two actions are tightly connected. As a minimum the following procedure should be followed:

1. Encode by coders 1 and 2.
2. Check and merge codings (performed by coders 1 and 2 until consensus).

**Creation notes:**
Authors: Hans Dybkjær and Laila Dybkjær.
Version: 1 (25 November 1998), 2 (19 June 1999).
Comments: For guidance on how to identify communication problems and for a collection of examples the reader is recommended to look at [Dybkjær 1999].
Literature: [Bernsen et al. 1998].

```
<u id="S1:7-1-sun" who="S">
  <w id="S1:7-1-sun-1">flight</w>
  <w id="S1:7-1-sun-2">information</w>
  british airways good day can I help you
</u>
<u id="U1:7-1-sun" who="U">
  #h yes I'm enquiring about flight number bee ay two eight six flying in
  later today from san francisco could you tell me %coughs% 'scuse me which
  airport and terminal it's arriving at and what time
</u>
<u id="S1:7-2-sun" who="S">
  please repeat your request
</u>
<u id="U1:7-2-sun" who="U">
  #h I'm enquiring about the the flight bee ay two eight six flying in later
  today from san francisco #h and I want to know which airport it's arriving
  at and which terminal and also what time hh please hh.
</u>
<u id="S1:7-3-sun" who="S">
  please wait
</u>
<u id="U1:7-3-sun" who="U">
  thank you
</u>
<u id="S1:7-4-sun" who="S">
  I'm sorry flight bee ay two six six does not leave from san francisco
</u>
<u id="U1:7-4-sun" who="U">
  I think there's been a mistake em I'm sorry I m- I meant bee ay two eight
```

```
  six hh bee ay two eight six yeh and it's the same information which airport
  does it arrive at and which terminal and what time
</u>
<u id="S1:7-5-sun" who="S">
  please wait...  flight two eight six from san francisco arrives at london
  heathrow terminal four at thirteen ten
</u>
<u id="U1:7-5-sun" who="U">
  thank you
</u>
<u id="S1:7-6-sun" who="S">
  goodbye
</u>
```

**Figure 3.** Orthographic transcription of dialogue 1:7 from the Sundial corpus. Only a very sparse kind of annotation is indicated. Markup of words is only shown for the two first words of the dialogue.

```
<u id="S1:9-1-sun" who="S">
  good afternoon british airways flight information
</u>
<u id="U1:9-1-sun" who="U">
  #h em I'm enquiring about bee ay nine oh three the
  flight from frankfurt to heathrow coming in tomorrow
  morning could you tell me what time it arrives, at heathrow
</u>
<u id="S1:9-2-sun" who="S">
  please wait... that flight is scheduled for twelve noon
</u>
<u id="U1:9-2-sun" who="U">
  right thank you very much
</u>
<u id="S1:9-3-sun" who="S">
  goodbye
</u>
```

**Figure 4.** Orthographic transcription of dialogue 1:9 from the Sundial corpus. Only a very sparse kind of annotation is indicated.

```
<vtype id="GG3-1" instance_of="Guidelines-1999#GG3">
  'flight information' known to be false: S knows only BA and only partially.
</vtype>
<vtype id="GG6-1" instance_of="Guidelines-1999#GG6">
  Not clear whether scheduled is only used about flight already departed and
  thus means expected, or whether it means according to the timetable.
</vtype>
<vtype id="GG7-1" instance_of="Guidelines-1999#GG7">
  Missing distinction between expected and timetabled arrival time.
</vtype>
<vtype id="SG2-1" instance_of="Guidelines-1999#SG2">
  No feedback on date.
</vtype>
<vtype id="SG2-2" instance_of="Guidelines-1999#SG2">
  No feedback on BA.
</vtype>
<vtype id="SG2-3" instance_of="Guidelines-1999#SG2">
  Missing feedback on flight and date.
</vtype>
<vtype id="SG3-1" instance_of="Guidelines-1999#SG3">
  Variation of system formulation.
</vtype>
<vtype id="SG4-1" instance_of="Guidelines-1999#SG4">
  Too little said on what system can and cannot do: BA often missing; time-table
  enquiries always missing.
</vtype>
<vtype id="SG5-1" instance_of="Guidelines-1999#SG5">
  Open S intro requires interaction instructions on waiting, verbosity etc.
</vtype>
```

**Figure 5.** Types of guideline violation detected in the two Sundial dialogues shown in Figures 3 and 4.

```
<comprob id="1" vtype="Sundial_violations#SG3-1" uref="Sundial#S1:7-1-sun"/>
<comprob id="2" vtype="Sundial_violations#GG3-1" wref="Sundial#(S1:7-1-sun-1 S1:7-1-sun-4)"/>
```

```
<comprob id="3" vtype="Sundial_violations#SG4-1" uref="Sundial#S1:7-1-sun"/>
<comprob id="4" vtype="Sundial_violations#SG5-1" uref="Sundial#S1:7-1-sun"/>
<note id="1" uref="Sundial#S1:7-1-sun">
  The system's introduction varies from dialogue to dialogue.
  This probably reflects a deliberate decision to test the effect
  on users of various system introductions.
  Still, in a working application this may cause confusion in users resulting in user
  questions which the system cannot understand.
  The core problem is that not all system introductions provide
  the same information or are equally informative.
  The designers must optimise the introduction rather than varying it.
</note>
<note id="2" uref="Sundial#S1:7-1-sun">
  The system provides too little information about its capabilities and limitations.
  It is of course an ideal that little information is necessary.
  However, the risk is that the user will be misled and assume stronger or weaker system
  capabilities than are actually present.
  Designers should look out for symptoms to this effect.
  The present introduction suggests that users can ask about anything to do with British
  Airways flights.
  No current system is likely to be able to do that.
  Another interpretation of the system's introduction is that it is owned by British Airways
  but can answer any question about flights.
  The former interpretation seems the most natural one.
  So the system's opening probably should not be deemed ambiguous.
</note>
<note id="3" uref="Sundial#S1:7-1-sun">
  The system provides no information on how to interact with it.
  It is of course an ideal that this should not be necessary.
  However, the risk is that the user will be misled and assume that one may interact with
  the system just like with a human operator, which is not possible today.
  Designers should look out for symptoms to this effect.
</note>
<note id="4" uref="Sundial#S1:7-4-sun">
  Misunderstanding of flight number.
</note>
<comprob id="5"  vtype="Sundial_violations#SG2-1" uref="Sundial#S1:7-5-sun"/>
<comprob id="6"  vtype="Sundial_violations#SG2-2" uref="Sundial#S1:7-5-sun"/>
<comprob id="7"  vtype="Sundial_violations#GG7-1" uref="Sundial#S1:7-5-sun"/>
<comprob id="8"  vtype="Sundial_violations#SG3-1" uref="Sundial#S1:7-6-sun"/>
<comprob id="9"  vtype="Sundial_violations#SG3-1" uref="Sundial#S1:9-1-sun"/>
<comprob id="10" vtype="Sundial_violations#GG3-1" uref="Sundial#S1:9-1-sun"/>
<comprob id="11" vtype="Sundial_violations#SG4-1" uref="Sundial#S1:9-1-sun"/>
<comprob id="12" vtype="Sundial_violations#SG5-1" uref="Sundial#S1:9-1-sun"/>
<comprob id="13" vtype="Sundial_violations#SG2-3" wref="Sundial#(S1:9-2-sun-1,S1:9-2-sun-2)"/>
<comprob id="14" vtype="Sundial_violations#GG6-1" wstart="Sundial#(S1:9-2-sun-3,S1:9-2-sun-4)"/>
<comprob id="15" vtype="Sundial_violations#SG3-1" uref="Sundial#S1:9-3-sun"/>
```

**Figure 6.** Annotation of communication problems in the two Sundial dialogues shown in Figures 3 and 4.

# 5. Reference and Markup of Communication Problems

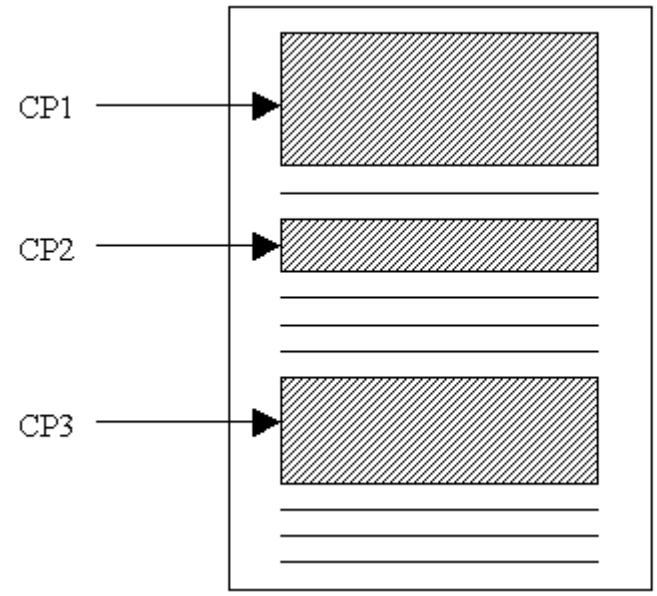
Communication problems span a wide range of possibilities as regards to scope of reference. They refer either to some item of information which was omitted, or to a single word, several words, a whole utterance, several utterances (or turns) or even in principle more than one dialogue which led to the miscommunication. In practice, communication problems most frequently refer to the first four of the options listed.

Furthermore, the markup of communication problems is not always non-contiguous but may overlap in various ways. This is described in the following and illustrated with textual examples through Figures 7-10.

*Non-Contiguous Tagging*

This is the most frequent kind of tagging for communication problems. Unlike other levels of discourse segmentation and tagging (such as morphological or prosodic for example), communication problems are by their very nature non-contiguous, i.e. not every utterance, nor every word will violate the guidelines for co-operative dialogue, and those will remain unmarked.

`<CP1>___</CP1>[___]<CP2>___</CP2>...`etc.

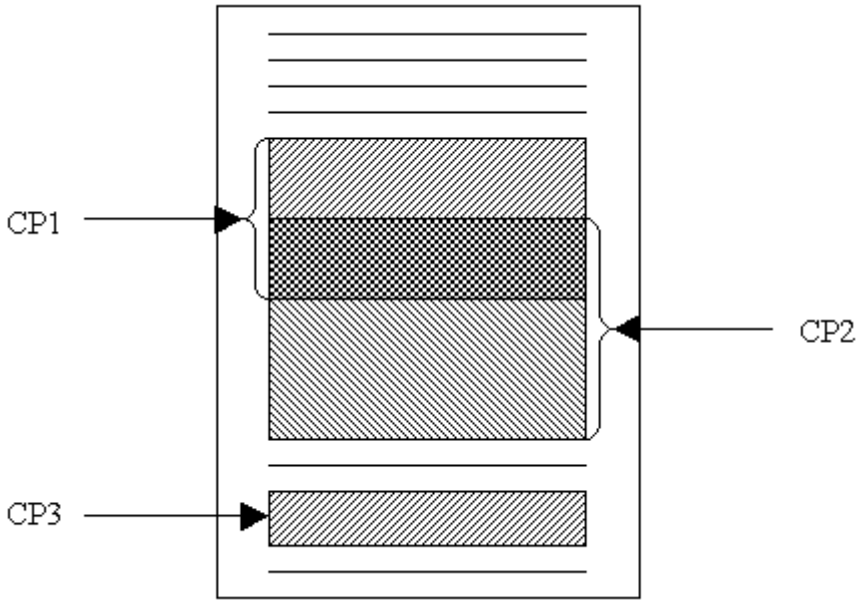



Examples are generally easy to find. In Figure 9 (below) CP3 and CP4 are non-contiguous.

*Overlapping Tagging*

In dialogues with many communication problems, such as early Wizard of Oz dialogues, overlapping tagging is quite frequently needed because overlapping parts of the same utterance often violate more than one guideline. This shows that, unlike in the case of part-of-speech and dialogue act tagging where each segment of a dialogue is tagged with one and only one tag, there is no one to one correspondence between words or utterances and their respective mark-up at the communication problems level. In this sense, overlapping tagging is very similar and bears a clear relation to nested and multiple tagging (described below) in that they all allow for the tagging of the same piece of text with more than one communication problem.

`<CP1>___<CP2>___</CP1>___</CP2>...`etc.

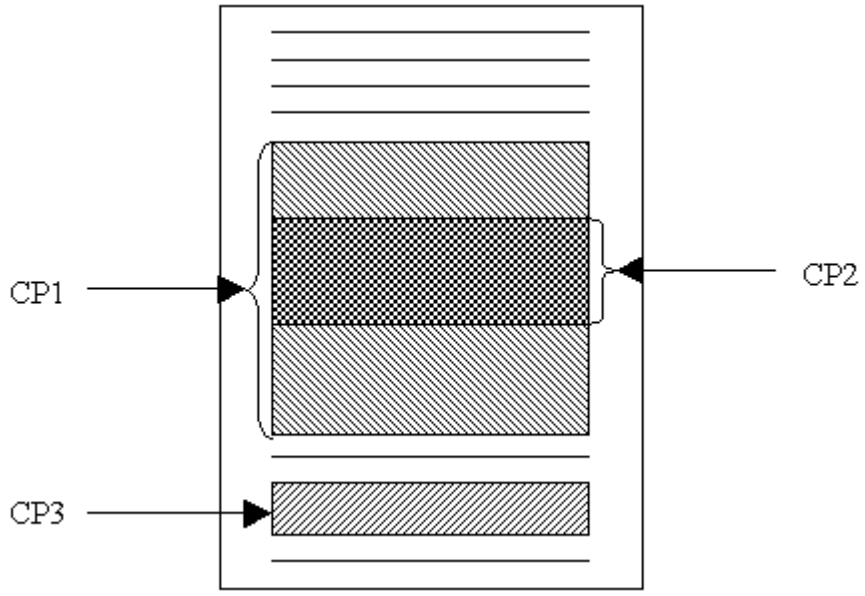



In Figure 9 there is a partial overlap between CP1 and CP2.

*Nested Tagging*

In nested tagging the part of a dialogue annotated as violating a particular guideline is totally embedded in and overlaps with a larger part of the dialogue annotated as violating a different guideline.

<CP1>___<CP2>___</CP2>___</CP1>...etc.

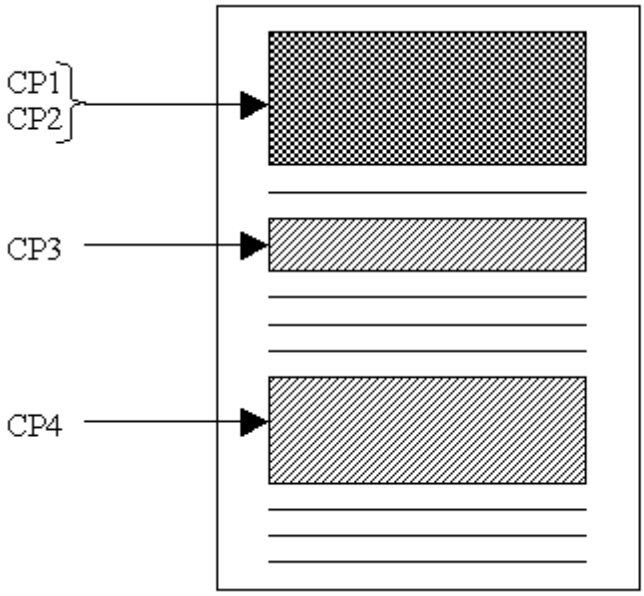



An example is CP7 and CP8, where CP8 is totally embedded in CP7, cf. Figure 9.

*Multiple Tagging*

Multiple tagging is a full overlap. Sometimes the same part of a dialogue violates more than one guideline.

<CP1><CP2>___</CP1></CP2>...etc.

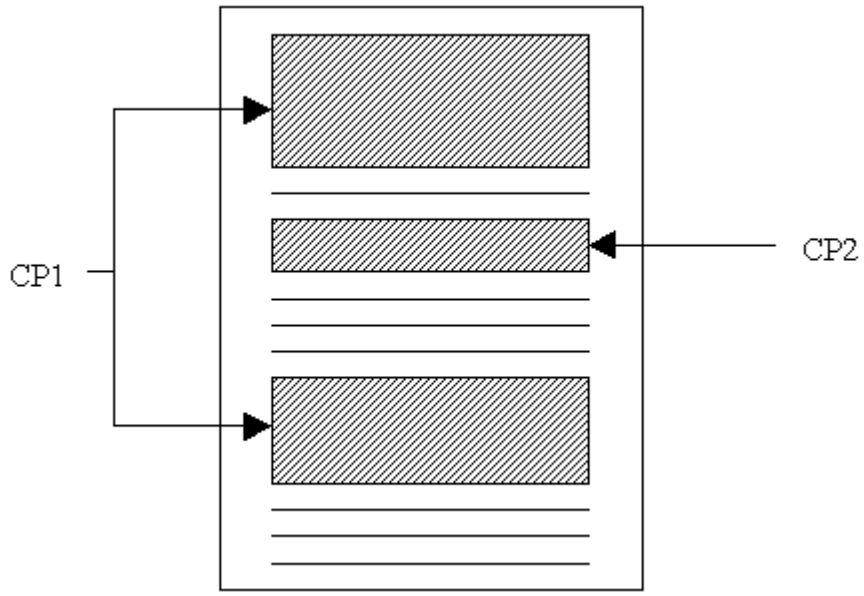


There is a full overlap between CP2 and CP3 in Figure 9, i.e. the same part of the dialogue is annotated twice. Please note, however, that CP2 refers to what is in the text (although it is insufficient) whereas CP3 refers to what is not there at all. One has to look at the whole utterance to conclude that something is missing. This is why the whole utterance it tagged in CP3.

*Discontinuous Tagging*

Discontinuous tagging where separate parts of the dialogue are annotated as part of one and the same communication problem does not occur very frequently. Normally communication problems refer to (part of) the same turn. Discontinuous tagging might e.g. be needed if the system is interrupted by the user or in order to relate a kind of explanation to the occurrence of a communication problem. The text between the two parts of the problem may or may not contain other communication problems.

<CP1>___</CP1>___ [<CP2>___</CP2>] ___<CP1>___</CP1>...etc.

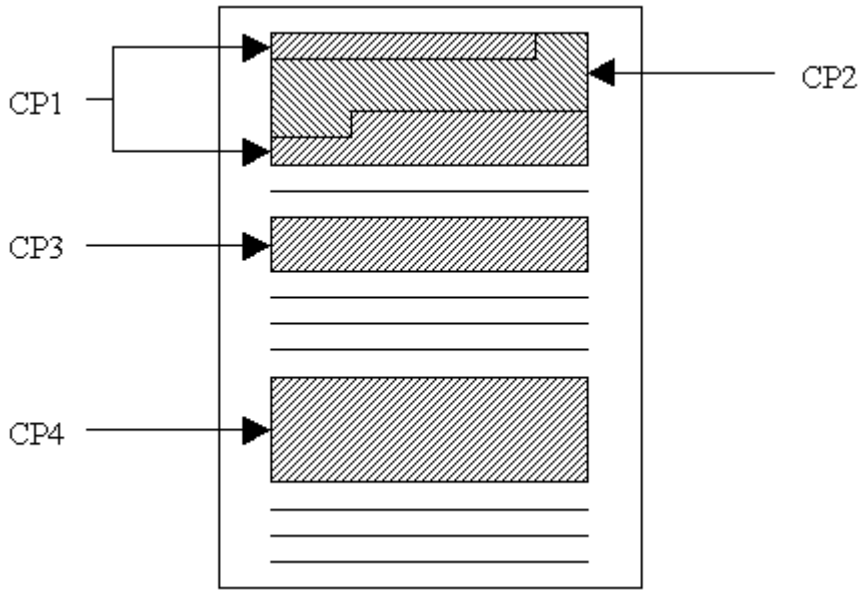



An example is CP4 together with the note N4 referring to S8:9-4 in Figure 9. CP4 annotates a violation in S8:9-2, but only the note N4 accompanying S8:9-4 makes it completely clear why CP4 is a violation of GG5. This relation is indicated in the note.

*One Turn Contiguous Discontinuous Tagging*

Sometimes a communication problem refers to discontinuous parts of an utterance. The text between the annotated parts may or may not contain other communication problems.

<CP1>___</CP1> [<CP2>___</CP2>] <CP1>___</CP1>...etc.




The translated snippet from the start of a dialogue with the Philips train timetable information system provides an example of discontinuous tagging within the same turn, cf. Figure 10.

```
<dialogue id="Dialogue_Sundial_8:9">
  <u id="S8:9-1" who="S">
    <w id="S8:9-1_1">flight</w>
    <w id="S8:9-1_2">information</w>
    can I help you (2.5)</u>
  <u id="U8:9-1" who="U">
    yeah good afternoon er:m (1.5) tomorrow (.5) eve- ahr yes tomorrow night (.5)
    (sa-em should be a flight arriving from warsaw (.5) er:m: it arrives (.5)
    I think a heathrow terminal two at around about half past nine (.7)
    could you confirm that please (5.5)
  </u>
  <u id="S8:9-2" who="S">
    please wait (37) I'm sorry
    <w id="S8:9-2_5">there</w>
    <w id="S8:9-2_6">are</w>
    <w id="S8:9-2_7">no</w>
    <w id="S8:9-2_8">british</w>
    <w id="S8:9-2_9">airways</w>
    <w id="S8:9-2_10">flights</w>
    <w id="S8:9-2_11">from</w>
    <w id="S8:9-2_12">warsaw</w>
    <w id="S8:9-2_13">please</w>
    <w id="S8:9-2_14">try</w>
    <w id="S8:9-2_15">polish</w>
    <w id="S8:9-2_16">airlines</w>
    (5)
  </u>
  <u id="U8:9-2" who="U">
    hm: (3) er:m (1) you sure of that (.)
    no british airways flights from warsaw to london (5.5)
  </u>
  <u id="S8:9-3" who="S">
    no (5.5)
  </u>
  <u id="U8:9-3" who="u">
    er:m (1) okay is there a flight that arrives (.5)
    at heathrow terminal two at nine thirty pee em tomorrow (4.5)
  </u>
  <u id="S8:9-4" who="S">
    please wait (31) flight pee oh two eight one from warsaw to london heathrow
    terminal two
    <w id="S8:9-4-16">arrives</w>
    <w id="S8:9-4_17">at</w>
    nine forty pee em (2.3)
  </u>
  <u id="U8:9-4" who="U">
    great (.5) that's it then (.5) thanks (3.5)
```

```
  </u>
  <u id="S8:9-5" who="S">
    thank you good bye
  </u>
</dialogue>
```

**Figure 7.** Transcription of dialogue 8:9 from the Sundial corpus with partial word markup.

```
<vtype id="GG3_VT1" instance_of="guidelines.xml#id(GG3)">
  The introduction announces a much broader domain coverage than the system
  actually possesses.
</vtype>
<vtype id="SG4_VT1" instance_of="guidelines.xml#id(SG4)">
  The system provides too little information about its capabilities and
  limitations.
  It is of course an ideal that little information is necessary.
  However, the risk is that the user will be misled and assume stronger or weaker
  system capabilities than are actually present.
</vtype>
<vtype id="SG5_VT1" instance_of="guidelines.xml#id(SG5)">
  The system provides no information on how to interact with it.
  It is of course an ideal that this should not be necessary.
  However, the risk is that the user will be misled and assume that one may
  interact with the system just like with a human operator, which is not possible
  today.
</vtype>
<vtype id="GG5_VT1" instance_of="guidelines.xml#id(GG5)">
  The system has announced that it can handle flight information in general.
  Therefore it seems irrelevant to tell the user that there are no British Airways
  flights from Warsaw and refer to Polish airlines.
  This was not what the user wanted to know, and apparently the system actually
  has the desired information, cf. utterance S8:9-4.
</vtype>
<vtype id="SG8_VT1" instance_of="guidelines.xml#id(SG8)">
  Since the system has announced that it can handle flight information in general,
  the user can rightly expect that the system can answer his question.
</vtype>
<vtype id="GG4_VT1" instance_of="guidelines.xml#id(GG4)">
  The system should never say something for which it does not have sufficient
  evidence.
</vtype>
<vtype id="SG2_VT1" instance_of="guidelines.xml#id(SG2)">
  No feedback on day. Is there a flight every day from Warsaw to London arriving
  at 9:40 pm?
</vtype>
<vtype id="GG1_VT1" instance_of="guidelines.xml#id(GG1)">
  The system does not explicitly distinguish between scheduled and actual arrival
  time.
  As there is no feedback on day there is nothing to clarify what is meant.
  In principle, the time provided may be the expected arrival time for the flight
  today and not the arrival time indicated in the timetable for a flight tomorrow.
</vtype>
```

**Figure 8.** Marked up types of guideline violations found in dialogue 8:9 from the Sundial corpus. In a dialogue (and in the full corpus) the same type may occur several times.

```
<comprob id="CP1" vtype="Sundial_violations.xml#GG3_VT1"
                                    wref=Sundial_trans.xml#(S8:9-1_1, S8:9-1_2)")/>
<comprob id="CP2" vtype="Sundial_violations.xml#SG4_VT1"
                                    uref="Sundial_trans.xml#S8:9-1"/>
<comprob id="CP3" vtype="Sundial_violations.xml#SG5_VT1"
                                    uref="Sundial_trans.xml#S8:9-1"/>
<comprob id="CP4" vtype="Sundial_violations.xml#GG5_VT1"
                                    wref="Sundial_trans.xml#(S8:9-2_5,S8:9-2_16)"/>
<comprob id="CP5" vtype="Sundial_violations.xml#GG8_VT1"
                                    wref="Sundial_trans.xml#(S8:9-2_5, S8:9-2_16)"/>
<note id="N1" uref=" Sundial_trans.xml#U8:9-2">
  Interestingly, the user questions the system's status as a perfect domain expert.
  This should be virtually inconceivable in an implemented system.
</note>
<note id="N2" uref=" Sundial_trans.xml#S8:9-3">
  The system is able to discuss its knowledge at meta-level.
  This is beside the point in current information systems who should be perfect domain
  experts.
</note>
<comprob id="CP6" vtype="Sundial_violations.xml#GG4_VT1"
                                     uref="Sundial_trans.xml#S8:9-3"/>
<note id="N3" uref=" Sundial_trans.xml#U8:9-3">
  This is basi ally the same question as in U8:9-1
```

```
</note>
<note id="N4" uref=" Sundial_trans.xml#S8:9-4">
  The system delivers a standard flight no. response package.
  Note that it turns out to know about flights other than British Airways.
  The user has no idea about the extent of the system's knowledge.
</note>
<comprob id="CP7" vtype="Sundial_violations.xml#SG2_VT1"
                                    uref="Sundial_trans.xml#S8:9-4"/>
<comprob id="CP8" vtype="Sundial_violations.xml#GG1_VT1"
                                    wref="Sundial_trans.xml#(S8:9-4-16,S8:9-4-17)"/>
```

**Figure 9.** Marked up communication problems found in dialogue 8:9 from the Sundial corpus. There are references to violation types in Figure 8 and to the transcription in Figure 7.

```
<u id="dir10:info254-S-1" who=S>
  You are connected to a speech recognition prototype made by Philips Research in Aachen.
  Hello.  This is the automatic train timetable information.
  You may ask about train connections to a thousand domestic railway stations.
  From where to where would you like to go?
</u>
```

```
<vtype id="GG8_VT1" instance_of="guidelines.xml#id(GG8)">
  The sentence "This is the automatic train timetable information." seems redundant since
  it has already been said that the user is connected to a system and in the succeeding
  sentence it is said what the system can do.
</vtype>
```

```
<comprob
  id="CP1" vtype="Philips_vtypes.xml#GG8_VT1"
  wref="Philips_trans.xml#(dir10:info254-S-1_1,dir10:info254-S-1_14),
                          (dir10:info254-S-1_16, dir10:info254-S-1_34)"
/>
```

**Figure 10.** Transcription, violation type and communication problem for part of a dialogue from the Philips corpus. Words are assumed to be marked up in the transcription although not shown here. However, a reference to words is used in the annotation of the communication problem.

The Communication Problem 'Domino' Effect

It is quite often the case that when the system violates one of the guidelines for co-operative communication, this will then prompt a user to respond inappropriately in turn, thus causing the system to make further errors, and so on creating a kind of chain of errors, or 'domino' effect.

We do not have any control over communication problems purely caused by the user. However, if a user's violation of a guideline is triggered by a previous system guideline violation the systems developer should certainly do something to avoid it and the problem should be annotated.

We have included the possibility of accounting for this phenomenon by using the attribute `caused_by` when marking up communication problems. If used, the attribute `caused_by` will point to some communication problem occurring earlier in the dialogue. Moreover, the `note` element may be used to provide an overall description of a chain of problems.

This domino effect of communication problems can clearly be seen in the dialogue below from the user test of the Danish Dialogue System for domestic flight ticket reservation, cf. Figure 11. The problem starts in S6-39a caused by insufficient grammar and vocabulary of the system. The user never succeeds in correcting the misunderstood date and eventually seems to give up and ignore new system misunderstandings although they are made clear through system feedback, e.g. in S6-44a. Thus the user ends up with a ticket which is quite different from the one s/he actually wanted.

```
        Dialogue T32a

[Scenario T-3-2-a]
S6-27b Do you want anything else?
```

```
U6-27a Yes.
S6-28a Customer number 4.
S6-28b Is that correct?
U6-28a No, customer number 3.
S6-29a Customer number 10 does not exist.
S6-29b Please state your customer number.
U6-29a Customer number 3.
S6-30a Customer number 3.
(overlap)  Yes.
S6-30b How many people will travel?
U6-30a Three people.
S6-31a Three people.
S6-31b Please state the id-number of the first person.
U6-31a Number 6.
S6-32a Please state the id-number of the second person.
U6-32a Number 3.
S6-33a Please state the id-number of the third person.
U6-33a Number 4.
S6-34a Id-numbers 6, 3 and 4, Anders Bækgaard, Paul Dalsgaard and Børge Lindberg.
S6-34b Where does the journey start?
U6-34a The journey starts in Aalborg.
S6-35a From Aalborg.
S6-35b Where does the journey end?
U6-35a In Copenhagen.
S6-36a In Copenhagen.
S6-36b Do you want a return ticket?
U6-36a Yes.
S6-37a Are you particularly interested in discount?
U6-37a Yes.
S6-38a On which date will the journey start?
U6-38a The first weekend in February.
S6-39a Friday February 10th.
S6-39b At which time?
U6-39a It must be on Saturday at 7:20.
S6-40a There is no departure at 7:20.  The closest other red departure is at 10:50.
S6-40b Do you want this departure?
U6-40a No.
S6-41a At which time?
U6-41a Saturday at 7:20.
S6-42a There is no departure at 7:20.  The closest other red departure is at 10:50.
S6-42b Do you want this departure?
U6-42a Yes.
S6-43a On which date will the return journey start?
U6-43a On Sunday February 5th.
S6-44a Sunday February 12th.
S6-44b At which time?
U6-44a 22:40.
S6-45a 22:40.
S6-45b You have now booked 3 tickets for Anders Bækgaard, Paul Dalsgaard and
       Børge Lindberg from Aalborg to Copenhagen on Friday February 10th at 10:50
       returning on Sunday February 12th at 22:40. The tickets are on green
       conditions.  Your reference number is 51.  The total price is 2820 kroner.
S6-45c Are the tickets to be mailed or will the travellers pick them up at the
       airport?
U6-45a They should be mailed.
S6-46a The tickets will be mailed about one week before the departure.
S6-46b Do you want anything else?
U6-46a Yes.
```

**Figure 11.** Dialogue T32a from the user test of the Danish Dialogue System. S is system and U is user.

## References

[Bernsen et al. 1996] Bernsen, N.O., Dybkjær, H. and Dybkjær, L.: Co-operativity in Human-Machine and Human-Human Spoken Dialogue. In *Discourse Processes*, Vol. 21, No. 2, 1996, 213-236.

[Bernsen et al. 1998] Bernsen, N.O., Dybkjær, H. and Dybkjær, L.: *Designing Interactive Speech Systems. From First Ideas to User Testing.* Springer Verlag 1998.

[Dybkjær 1999] Dybkjær, L.: CODIAL—a Tutorial and Tool in Support of Co-operative Dialogue Design. http://www.nis.sdu.dk/~laila/dialogue.

[Dybkjær et al. 1998] Dybkjær, L., Bernsen, N.O., Dybkjær, H., McKelvie, D. and Mengel, A.: The MATE Markup Framework. MATE Deliverable D1.2, November 1998.

[Grice 1975] Grice, P.: Logic and conversation. In P. Cole and J. L. Morgan (Eds.), *Syntax and Semantics* Vol. 3: *Speech Acts*. New York: Academic Press 1975, 41-58. Reprinted in Paul Grice: *Studies in the Way of Words*. Cambridge, MA, Harvard University Press, 1989.

[Klein et al. 1998] Klein, M., Bernsen, N.O., Davies, S., Dybkjær, L., Garrido, J., Kasch, H., Mengel, A., Pirrelli, V., Poesio, M., Quazza, S. and Soria, S.: Supported Coding Schemes. MATE Deliverable D1.1, July 1998.

# Cross Level Annotation

Andreas Mengel

## 1. Coding Purpose

This chapter describes means for the encoding of cross-level annotation. *Cross-level* in this context is meant to denote all kinds of phenomena characterized by reference to other elements. Thus, all concepts described here do not necessarily originate from existing theoretical concepts of a given linguistic theory, but are meant to provide the annotator with theory-independent tools necessary for the additional annotation of existing corpora represented in XML.

*Cross-level* refers to relations that can be established between any two or more elements. These elements may belong to different types of elements, e.g., `<word>` and `<phone>`, but may also belong to the same element type e.g., `<sentence>` and `<sentence>` (e.g., word order phenomena or lexical relations). Thus, *cross-level annotation* refers to any kind of markup that links any two or more XML elements together.

## 2. Existing Schemes

Schemes looked at for an evaluation [1] of existing cross-level markup were the following:

| | |
|---|---|
| BAS | Bavarian Archive for Speech Signals |
| BNC | British National Corpus |
| CHI | Child Language Data Exchange System |
| DAM | Dialogue Act Markup in Several Layers |
| KCS | Kiel Corpus of Read and Spontaneous Speech |
| SAB | Synthesis Markup Language |
| SAM | SAM Standards |
| TEI | Text Encoding Initiative |
| TRA | Natural Spoken Dialogue and Interactive Planning |
| VER | Verbmobil II Conventions for Spontaneous Speech |
| BFR | Bonn Focus Research |

A number of criteria were used to assess the applicability of individual schemes and it showed that - as most of them were designed for special theoretical frameworks and can only be read by specialized parsers - the adaptability to more general purposes and cross reference between different markup data was very limited.

## 3 Selected Scheme(s)

As reported in the [1], a markup grammar that uses concepts like XML is most powerful. This is why the cross-level markup uses standard mechanism of reference as provided by XML instead of layout oriented or multicolumn formats used by other markup conventions. The following will address principles and assumptions on which the MATE cross-level markup are based.

### 3.1 Cross-level annotation requirements

There are a couple of theoretical and formal requirements needed for cross-level annotation.

Basic requirements:

- *Elements*: It is obvious that the first basic requirement for corss-level annotation is that there is preferably more than one element or even more than one level of annotation (i.e. more than one element type) available. Of course, the number of elements, one or more, does not depend on the number of units physically present, but on the number of different descriptions that are attached to it (i.e., the number of elements). Thus, even a single word can be described with respect to many different aspects: Its orthographic representation, the sequence of sounds, their spectrographic properties etc.

- *Common ground*: Having more than one level of description is not sufficient for the linking of different elements; an infrastructure for symbolic reference from one to another is mandatory. This common infrastructure is present if it is made sure that the representations of the elements to be linked can be related to each other. Therefore, these elements must be XML-elements. This means that obviously only items from existing XML documents can be referred to and that these items can only be (XML) elements. There is no possibility to refer to attributes or values of element attributes.

Conceptual requirements:

- *Theories*: Everything can be linked to everything else, but then, there is no information in a link. Thus, there must be theories or questions about the interrelation of elements and its communicative or theoretical relevance. These questions can then be reformulated into identification strategies of queries applied to the data.

- *Identification process*: Linking any elements together does not make sense and the aim of annotation is the identification of an algorithm, an operationalisation of what the properties of the elements to be marked and linked are. Thus, in an ideal world, elements marked up by cross-level annotation should have a relation that can be described formally. Hence, cross-level annotation does not require that an algorithm is involved in the process of annotation, but this might be an aim. Even if it is not possible for the annotator to formally describe the properties that make up the relation between elements combined, there might be theories and descriptions for the elements that enable a formal description of the relations. Thus, many relations between any two ore more elements to be linked can be expressed by a query language (e.g., Q4M [3]): As query languages like Q4M state relations betweens elements and as it should be possible to describe (and mark up) the properties of elements that (theoretically grounded) links are referring to, there is a strong correlation between the output of a query language and cross-level annotation. If all information needed to describe the relations of elements in a given cross-level environment are known and marked up and if the query language used offers all the operators required to express these relations, then the process of cross-level annotation can be executed by using that query language for the specification of the relations and by using the query processor for the production of the markup.

  One implication of this is, that relations that cannot be expressed formally, lack theoretical knowledge or markup in the corpora. The extensional definition of a cross-level phenomenon (a relation between elements) is the cross-level annotation which can be the result of a query, one possible representation of the intensional definition of a cross-level relation is a query expression.

Representation requirements:

- *Accessibility:* The elements referred to must be accessible, i.e. they must have an attribute of type ID.
- *Uniqueness:* The reference to the elements referred to must be unique, i.e. the combination of the filename and the ID of the elements has to be unambiguous within the environment where the cross-level document is used.

## 3.2 Principles

The range of possible applications of cross level annotation covers theoretically driven tasks as well as problems of practical corpus preparation (see above). The problem is, that these applications imply that virtually anything needs to be linkable, either to a new element, or to another existing element, or to sets of other elements. And, since these new links can neither have a predefined structure nor semantics derived from existing level-specific coding schemes, general and universally applicable principles for the annotation of cross-level tags have to be developed. Only then will linking of elements of any type be possible.

Cross-level annotation is perceived as new information being added to existing markup of linguistic data. Thus, any cross-level annotation is a new level of annotation and will use element names different from the elements referred to. Any cross-level annotation is referential, i.e., every `<xlr>` element of a cross-level document will refer to existing elements of other XML documents. In this respect, cross-level annotation is very conservative, too: As there will be new elements for cross-level annotation, existing annotation and associated files will not be changed and this will make it easier to distinguish between different levels of annotation.

As there is a strong connection between cross-level annotation and query processing, cross-level annotation should be capable of representing the output of queries, too.

### 3.2.1 Linking

The technical means to link between elements in XML is the href attribute which specifies the location (identity) of one or more elements that are to be linked to that element. Yet, the type of link, i.e. its functional meaning may differ. In general, 'links' are a very abstract way to express relations between information units.

#### 3.2.1.1 Relations

The table below shows techniques used for expressing various relations common to XML documents.

| relation | expressed by | example | recommended |
|---|---|---|---|
| part-whole | sub-element | `<word><phone/></word>` | yes |
| part-whole | href | `<word href="phon.xml#id(123)/>` | yes |
| attribute | XML-attribute | `<book title="Ulysses"/>` | yes |

| | | | |
|---|---|---|---|
| value | XML-value | `<book title="Ulysses"/>` | yes |
| attribute | sub-element | `<book><title>Ulysses</title></book>` | no |
| value | PCDATA | `<book><title>Ulysses</title></book>` | no |
| hyponomy | sub-element | `<word><noun/></word>` | no |
| sequence | sequence | `<word/><word/>` | yes |
| sequence | attribute | `<word n="001"/><word n="002"/>` | |
| content | attribute | `<word lem="house"/>` | yes |
| content | PCDATA | `<word>house</word>` | |

### 3.2.1.2 href attribute

In order to use an XML technique which is universally available, i.e. can be used to express a relation between any two elements, the href attribute should be exploited [2]. Consider the following example as a default situation. There is a file specifying the segments of an utterance.

pho.xml

```
...
<pho id="pho_01" type="t"/>
<pho id="pho_02" type="r"/>
<pho id="pho_03" type="I"/>
<pho id="pho_04" type="k"/>
<pho id="pho_05" type="w"/>
<pho id="pho_06" type="O:"/>
<pho id="pho_07" type="z"/>
...
```

(corresponding DTD: pho.dtd)

And the general case could be that there is also a markup of words using `<w>` elements that refer to these previously defined phones `<pho>` elements.

word.xml

```
...
<w id="w01" href="pho.xml#id(pho_01)..id(pho_04)/>
<w id="w02" href="pho.xml#id(pho_05)..id(pho_07)/>
...
```

(corresponding DTD: word.dtd)

### 3.2.1.3 Types links

Yet, there are cases, where one does not only want to link one element or a series of elements of one type to another but elements from different levels (and element types) to an element, e.g.

- link corefering words/expressions together, link them to a real-world referent
- link definitions to elements
- link word tokens to lexical elements

For information shared my many tokens of one type, there could be a kind of lexicon file that specifies information which is shared by many elements, so it is more efficient to specify this information once only:

wordlex.xml

```
...
<lemma id="lem_23233" pos="NN" pron="trIk" orth="trick"/>
<lemma id="lem_99887" pos="AV" pron="wO:z" orth="was"/>
...
```

(corresponding DTD: wordlex.dtd)

Now, a link must be established that connects individual <w> elements to their <lemma> type elements. As the href attribute inside the <w> elements is already taken, children elements of the <w> elements must be used to specify links to these <lemma> elements:

word2.xml

```
...
<w id="w01" href="pho.xml#id(pho_01)..id(pho_04)>
  <lexspec id="lexspec_02" href="wordlex.xml#id(lem_23233)"/>
</w>
<w id="w02" href="pho.xml#id(pho_05)..id(pho_07)>
  <lexspec id="lexspec_03" href="wordlex.xml#id(lem_99887)"/>
<w/>
...
```

(corresponding DTD: word2.dtd)

These XML children (of <w> elements) hold links to other elements and their element name (e.g., <lexspec>) specifies the kind of information that is provided by the link element.

Thus, a more generalized recommendation is to use hrefs in XML children for linking to elements which are functionally or conceptually motivated. These linking XML children (like <lexspec>) are XML elements the name of which can be used to specify the type of reference/link and the href attribute value of which points to an element.

Using embedded elements to specify relations among elements makes it possible to more explicitly specify the roles of the elements in that particular relation. If this information were only put into a TEI <link> [4] elements, then only knowledge about the specific order of the elements linked to can be exploited to provide information about the role within the relation. Of course, special attributes could be used to encode this information, too, but this would require extra parsing of the attributes and their values.

### 3.2.1.4 Linking desing criteria

The following decision scheme presupposes that links between elements that have a part-whole relationship - such as word and sentences - are a trivial case which can be represented by href attributes from the elements that represent the 'whole' concept to the elements which represent the 'part' concept (cf. *3.2.1 Linking*). The following decision tables are to provide a procedure for determining how links between any two ore more different element types are to be represented in XML.

| | | |
|---|---|---|
| Q1 | *There is a set of elements that have a relation to each other and these elements are not always pairs of direct neighbours and they are not always in a direct parent-children element relationship, either?* | |
| | • yes | Go to Q2 |

| | | |
|---|---|---|
| | • no | Go to E0 |
| Q2 | *How many elements do participate per relation?* | |
| | • only two | Go to Q3 |
| | • more than two | Go to Q4 |
| Q3 | *Do the elements have the same role in the relation?* | |
| | • yes | Go to E2 |
| | • no | Go to E1 |
| Q4 | *Do the elements have the same role in the relation?* | |
| | • yes | Go to E2 |
| | • no | Go to E3 |

| | |
|---|---|
| E0 | • No action required |
| E1 | • One of the elements <aelm> contains a newly to be created direct child element (<celm>) which has an href attribute that points to the other element (<belm>).<br>`...`<br>`<aelm id="aelm_001>`<br>`  <celm id="celm_001" href="belm.xml#id(belm_001)"/>`<br>`</aelm>`<br>`...`<br>(corresponding DTD: e1.dtd) |
| E2 | • A new element type (<celm>) is created. It contains newly to be created elements of type <delm> which are embedded children elements. These have an href attribute which points to the elements.<br>`...`<br>`<celm id="celm_001>`<br>`  <delm id="delm_001" href="aelm.xml#id(aelm_001)"/>`<br>`  <delm id="delm_002" href="aelm.xml#id(aelm_002)"/>`<br>`</celm>`<br>`...`<br>(corresponding DTD: e2.dtd) |

<table>
<tr><td rowspan="3">E3</td><td>
<ul>
<li>A new element type (<code><celm></code>) is created. It contains to be created elements of type <code><delm></code> which are embedded children elements. These have an href attribute which point to the elements. In order to indicate their different roles, they either have
<ul>
<li>a special attribute with different value.</li>
</ul>
<u>or</u>
<ul>
<li>different element types.</li>
</ul>
</li>
</ul>
</td></tr>
<tr><td>
<pre>
...
<celm id="celm_001>
  <delm id="delm_001" rel="e"
                      href="aelm.xml#id(aelm_001)"/>
  <delm id="eelm_001" rel="f"
                      href="aelm.xml#id(aelm_002)"/>
</celm>
...
                          or
...
<celm id="celm_001>
  <eelm id="eelm_001" href="aelm.xml#id(aelm_001)"/>
  <felm id="felm_002" href="belm.xml#id(belm_001)"/>
</celm>
...
</pre>
</td></tr>
<tr><td>(corresponding DTD: e3.dtd)</td></tr>
</table>

### 3.2.1.5 Link direction

In order to derive some abstract principles of the direction on href links, one could state the following:

- links are rather made from less reliable elements to more reliable ones (physical phenomena, perceptual phenomena)
- links are made from conceptually higher (i.e. more abstract) elements to lower ones (coreference element to words)
- links lead from elements produced later to those made earlier (from sentence to word)
- links are used to make important information to be accessible from any referring element (from token to type)

According to different purposes of cross-level annotation, the same syntactical link structures may have many different meanings. Above all, cross-level annotation is meant to provide general purpose XML structures that do not constrain their application to specific linguistic theories. However, this view does not exclude the application of cross-level annotation to theory driven annotation. In contrary, this makes the set of guidelines open for any annotation that is based upon existing XML documents.

Possible applications for cross-level annotation are listed here:

- theoretical
    - relate: associated phenomena are linked together

- new phenomena: annotation of (linguistic) phenomena that refer to existing annotation(s) of linguistic phenomena

- meta annotation
  - alternative annotation: for elements in a given annotation, alternative annotation elements are specified
  - annotation evaluation (qualitative): elements specifying the quality or correctness of a given annotation are produced
  - consistency check (structural): elements referring to elements that do or do not have (a given set of) structural properties defined as well formed.

### 3.2.2 Distribution of properties

The order of elements of the same type is a basic kind of linking elements which originally was meant to encode the proximity of the entities that are represented by these elements, e.g. sequences of words. If elements of different files are linked by href attributes then a specific relation is encoded this way. And, as the elements linked share a relation, there must also be some properties that are constitutive for this relation. In the standard way of linking information between entities in a part-whole relation this constitutive property is the extension in time: The `start` value of the first word in a sentence and the `end` value of the last word of a sentence are the same as the `start` and `end` value of the sentence they are part of. As this information is constitutive for this relation it can also be exploited for minimizing the redundancy and maintainability of data, thus one should represent `start` and `end` information only on the lowest level within a hierarchy that is defined by the part-whole relationship.

From a more abstract point of view one could describe attributes and the question of information infrastructure in the following way:

| **the attribute is applicable to ...** | | |
|---|---|---|
| more than one element type | one element type only | **value can...** |
| *general value* | *type value* | be inherited |
| `speaker identity`<br>`situation`<br>`start`<br>(by neighbour or part-whole relation) | `part of speech`<br>`case`<br>(by type-token relation) | |
| *local value* | *category value* | not be inherited but derived |
| `speech rate`<br>`duration`<br>(by application of a procedure) | `f0 slope type`<br>(by application of a procedure) | |
| *unique value* | *situation value* | neither be inherited nor derived |
| `element Ids` | `co-referent` | |

*Derivation* in this context denotes that a given value can be calculated by a procedure applied to other values.

Inheritance will is possible with the application of style sheets in the MATE environment.

How might a general strategy for these cases look like? It is obvious that attributes like the speaker identity, the coder etc. apply to more than one element or more than one element type even. So, it would be redundant to specify this information for every element. A principle one could state is to put general information at one special position. Now, it would seem natural that this position - e.g. in the case of the speaker identity - is the highest element in a hierarchy for which and for all children elements of which this information is true. Thus, information not explicitly defined for one element can be found by going up in a hierarchy.

On the other hand, there is the principle of separate files for different levels of description, which - in the case where two or more levels are in a hierarchical relation - requires href linking and thus remote embedding of the elements of these levels. The problem here is that in each higher level, information is available about elements and files of the lower level(s) - as there are href links to them - but not the other way round: There is no information in a `<word>` element that it has parent elements that refer to it unless the parent element file is loaded into a software package together with the `<word>` level file. Thus, a proposal different from that made above is, to put information of the speaker etc., i.e. information applicable to more than one element on different levels to the first element of the lowest level of a hierarchy to make this information available from any level. This is also sensible form a production point of view. The lowest level in the hierarchy would or should be the first level of transcription, thus on this stage of production of the document, all relevant information is recorded and must be available to any further annotation. If this information is already at hand, it can remain in the first annotation level and no complicated decision process for moving the information to the currently highest level once this is produced is required.

Only when following this idea, will it be sure that important information can always be tracked because the elements in which this information is not made explicit, have links to others. If it were the other way round, how would it be possible to know in which higher level file some special information can be found when there is no link?

Yet, to make this principle work, some more structure is necessary: Every attribute type has a domain of applicability, so the annotator has to know what this domain is. A procedure for this could be the following:

**Preparation of data**

- Identify all element types that can have this attribute
- For all of the element types define in the DTD that an element of this type can have this attribute
- Whenever an element has an attribute defined in the DTD and the value of that attribute is not specified in the markup of that element a procedure must be executed to find this information. Any locally specified information overrides distant (central) information
- Values of attributes which are applicable to more than one element (or element type even) should - for reasons of easy maintenance - be represented only once in a corpus.

For the placement and retrieval of this information, the following rules are stated:

**Placement of general attributes**

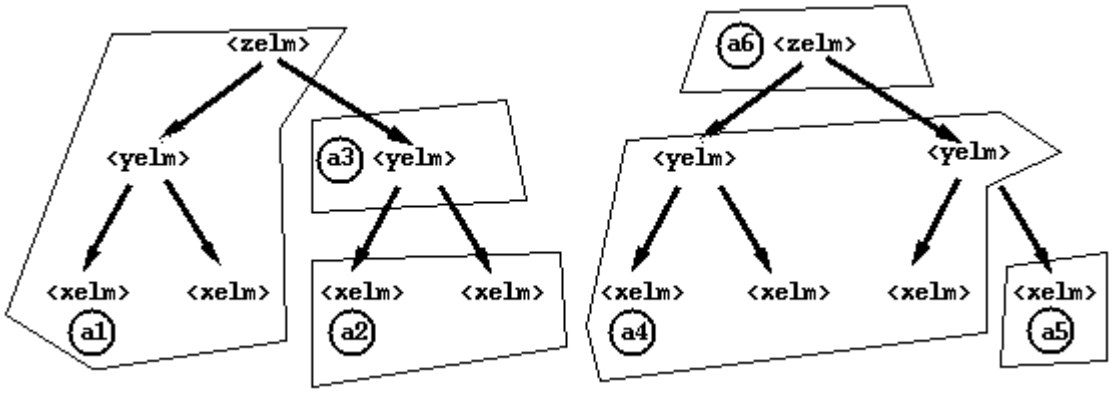
- In the first/lowest level of elements produced put the attribute to the first element of that level.
- Whenever the value of the attribute is not valid any more, place a new attribute specification (and the new value) into the markup of the first element for which this is necessary.
- For any (higher) level elements that refer to (these) existing other elements, assign the attribute and value if the first lower level element you encounter when searching for a value specification of that attribute has a value that is not applicable.

If the element can have a value of a given attribute (stated in the DTD) but there is no value specified in the element itself, retrieval procedures must be started.

**Retrieving values of general attributes**

- If the element has no child element go back in the row of element neighbours and take the value of the first element in the sequence of elements of this type where this value is specified.
- If the element has a child element, go to the first child element of this element. If the value is not specified here, proceed with the step before.

The following figure shows a group of elements of different type (`<xelm>`, `<yelm>`, `<zelm>`) and their reference structure. Additionally, the values `a1` to `a6` (all of the same attribute) have been specified for some of the elements. The regions show the application range of these values.




## 3.3 Applications

This and the following sections are dedicated to a number of possible applications of cross-level annotation. These sections provide examples of how the principles described before can be applied to a set of level independent purposes. They are divided into *Concurrent tagging, Resource annotation,* and *Annotation process*.

### 3.3.1 Concurrent tagging

In the context of annotating corpora, there may be situations where different versions of tagged data exist. Concurrent tagging information can occur in various situations:

- *Normalized version*: There is a standard transcription for a given utterance. An example for this might be an orthographic text that many speakers have read or a version of an utterance that is grammatically more acceptable. The different variants produced by different speakers should refer to this information.
- *Discontinuous constituents*: There is a level of description which refers to a lower level and the higher level elements point to one or more lower level elements but there are cases where the lower level elements belonging to one higher level element do not make up a continuous row of elements.
- *Multiple versions*: There is more than one tagging file produced by different coders related to the same phenomenological level.
- *Alternatives*: For some elements a coder wants to state that he or she could have decided differently so there is an option.

These cases are quite different and require specific treatment in annotation.

#### 3.3.1.1 Normalized version

In the normalized version case it has to be assumed that the normalized version is a standard, a reference the people which tag the - deviating - variants of individual speakers must refer to. Thus, in this case the normalized version is a kind of basic level which the newly to be produced tag levels (i.e. the transcription of actual utterances of the speakers) refer to. The difference to a tagging situation like the production of sentence annotation upon the word level is that the normalized version case will require a one-to-one mapping of source elements (assumed words) and referring elements (actually produced words) whereas in the sentence markup case, there is a one-to-many relation between the sentence elements and the word elements.

(Depending on the special situation presented here, in the DTD, the "show" axis will be set to "replace" and the "actuate" axis should be set to "user", that means that only if the user wishes to, the actual element that has a link to another (normalized version) element is replaced by this element pointed to.)

Consider a normalized version of a dialogue as a standard document upon and relative to which new documents have to be created, such as in the case of a normalized transcription of a given text which was read aloud by different people and where their utterances were recorded. Assume, that a part of the document is the normalized version of the word *mist*, i.e. the normalized utterance is `[mIst]` and its representation in XML might be:

normpho.xml

```
...
<pho id="pho001" type="m"/>
<pho id="pho002" type="I"/>
<pho id="pho003" type="s"/>
<pho id="pho004" type="t"/>
...
```

(corresponding DTD: normpho.dtd)

In the following section, four cases of deviation and proposals for their annotation will be described, namely replacement, deletion, insertion, and new order. Note that the scenario is such that there is the normalised/expected version (normpho.xml) and a number of utterances which are annotated referring to the standard.

### Replacement

The actual utterance of speaker A might be [mUst], a replacement (of one segment) which is the most simple case, thus, the XML representation of the utterance would be

replace.xml

```
...
<pho id="pho0090"          href="normpho.xml#id(pho001)"/>
<pho id="pho0091" type="U" href="normpho.xml#id(pho002)"/>
<pho id="pho0092"          href="normpho.xml#id(pho003)"/>
<pho id="pho0093"          href="normpho.xml#id(pho004)"/>
...
```

(corresponding DTD: replace.dtd)

So, all that is needed, is the addition of the href attribute and a different value in the type attribute of the second <pho> element.

### Deletion

The actual utterance of speaker B might be [mIt], a deletion of one segment. In that case, the XML representation of the utterance should be

delete.xml

```
...
<pho id="pho0274"         href="normpho.xml#id(pho001)"/>
<pho id="pho0275"         href="normpho.xml#id(pho002)"/>
<pho id="pho0276" type"-" href="normpho.xml#id(pho003)"/>
<pho id="pho0277"         href="normpho.xml#id(pho004)"/>
...
```

(corresponding DTD: delete.dtd)

Thus in this case, one can either specify a special character as a deletion symbol ("-") or make no reference to the element deleted at all (i.e., leave out the <pho> element with id="pho0276").

### Insertion

The actual utterance of speaker C might be [mIsIt], the insertion of a segment. In this case the XML representation of the utterance should be

insert.xml

```
...
<pho id="pho3027"          href="normpho.xml#id(pho001)"/>
<pho id="pho3028"          href="normpho.xml#id(pho002)"/>
<pho id="pho3029"          href="normpho.xml#id(pho003)"/>
<pho id="pho3030" type="I" href="normutt.xml#id(pho004)"/>
<pho id="pho3031"          href="normpho.xml#id(pho004)"/>
...
```

(corresponding DTD: insert.dtd)

Thus in this case, there are two elements pointing at the same element. Ideally, all elements which appear in addition, are mapped to the element to which the element behind also points to. In the case, where there is an additional element at the end of an utterance, it must point to the last element. Repetitions can be handled in an equivalent fashion.

**Different order**

The actual utterance of speaker D finally might be [simt], a new order of the segments. In this case the XML representation of the utterance should be

neworder.xml

```
...
<pho id="pho7997" href="normpho.xml#id(pho003)"/>
<pho id="pho7998" href="normpho.xml#id(pho002)"/>
<pho id="pho7999" href="normpho.xml#id(pho001)"/>
<pho id="pho8000" href="normpho.xml#id(pho004)"/>
...
```

(corresponding DTD: neworder.dtd)

In phonetic transcription, a case like *new order* is rather rare and only clearly identifiable if systematic new order phenomena (e.g. in second language acquisition) can be assumed. For levels like syntax, it might be easier to identify cases like this and distinguish them from simultaneous replacements of two or more elements.

### 3.3.2 Discontinuous constituents

The normal case of linking to hierarchically lower elements when adding a new higher level of description suggested here is using the `href` attribute the value of which is the name of the file and the id of the element to be linked to or the files and a list reference as in the following example:

*The thing suddenly spoke.*

normword.xml

```
...
<w id="w_001">The</w>
<w id="w_002">thing</w>
<w id="w_003">suddenly</w>
<w id="w_004">spoke</w>
...
```

(corresponding DTD: normword.dtd)

normsent.xml

```
...
<s id="s_001" href="normword.xml#id(w_001)..id(w_004)"/>
...
```

(corresponding DTD: normsent.dtd)

In the next example, it is not possible to use the list feature of the `href` attribute like in the the previous example because the elements are not all direct neighbours.

*The thing - no you cannot open the door now - suddenly spoke.*

normword2.xml

```
...
<w id="w_001">The</w>
<w id="w_002">thing</w>
<w id="w_003">no</w>
<w id="w_004">you</w>
<w id="w_005">cannot</w>
<w id="w_006">open</w>
<w id="w_007">the</w>
<w id="w_008">door</w>
<w id="w_009">now</w>
<w id="w_010">suddenly</w>
<w id="w_011">spoke</w>
...
```

(corresponding DTD: normword2.dtd)

If markup on a higher level refers to a set of elements that cannot be addressed by a list reference like in "`id(w_001)..id(w_004)`", another means of linking to lower level elements is required:

discont1.xml

```
...
<s id="s_001">
  <wlink id="wlink_001" href="normword2.xml#id(w_001)"/>
  <wlink id="wlink_002" href="normword2.xml#id(w_002)"/>
  <wlink id="wlink_003" href="normword2.xml#id(w_010)"/>
  <wlink id="wlink_004" href="normword2.xml#id(w_011)"/>
</s>
...
```

(corresponding DTD: discont.dtd)

or

discont2.xml

```
...
<s id="s_001">
  <wlink id="wlink_005" href="normword2.xml#id(w_001)..id(w_002)"/>
  <wlink id="wlink_006" href="normword2.xml#id(w_010)..id(w_011)"/>
</s>
...
```

(corresponding DTD: discont.dtd)

In the corresponding DTD, the "show" aspects would have to be set to "replace" in order to have the same effect as the href attributes in the examples above:

```
...
<!ATTLIST wlink
          href     CDATA  #IMPLIED
          xml:link CDATA  #FIXED    "simple"
          actuate  CDATA  #FIXED    "auto"
          show     CDATA  #FIXED    "replace"
          actuate  CDATA  #FIXED    "auto">
...
```

### 3.3.3 Multiple versions

In the case of multiple versions, there will be one common base level for which a higher level annotation is to be produced by different annotators. E.g. the orthographic transcription is

available and dialogue act annotation is provided by different coders. Thus, in this case there might be differences among the different annotations on the level of segmenting, too.

(Again, this requires the normal "href" attribute to the conceptual children. In the DTD, the "show" axis will be set to "embed", the actuate axis to "auto".)

### 3.3.4 Alternatives

There will be cases where the annotator wants to provide alternative solutions of grouping the lower level elements or of specifying values. One way to represent alternatives is to generate a new file for each alternative, such that for all combinations of all alternatives there is one file including the complete label information, since there is no possibility to use two value specifications for one attribute in one element e.g.

```
*<w pos="NN" pos="NA">
```

The problem with this option is that one would need $o = n_1 * n_2 * ... * n_i * n_k$ (with $k$ denoting the $k^{th}$ choice conflict and $n$ specifying the number of alternative choices of that value) different level files to represent all different combinations of alternatives.

Another - and recommended - way of representing alternatives is to provide the standard, preferred or most likely variant in the tag file. In addition to that, there will be one or more documents produced which contain alternative tagging information. These elements need at least two attributes: href and id. Additionally, all that information for which alternatives are possible is specified.

To give an example, first of all there is the preferred version:

normw.xml

```
...
<w id="w_234" who="pt" href="phon.xml#id(phon_824)..id(phon_825)>Is</w>
<w id="w_235" who="pt" href="phon.xml#id(phon_826)..id(phon_827)>it</w>
<w id="w_236" who="pt" href="phon.xml#id(phon_828)..id(phon_829)>the</w>
<w id="w_237" who="pt" href="phon.xml#id(phon_830)..id(phon_832)>right</w>
<w id="w_238" who="pt" href="phon.xml#id(phon_833)..id(phon_835)>thing</w>
...
```

(corresponding DTD: normw.dtd)

Now, consider it is not absolutely certain whether the first word was really an "right" or rather a "write" and there is doubt whether the last word was spoken by speaker "`pt`", as it might also have been speaker "`mr`":

altern.xml

```
...
<alt id="wa_001">Was
  <orig id="orig_001" href="normw.xml#id(w_237)"/>
</alt>
<alt id="wa_002" who="mr">
  <orig id="orig_001" href="normw.xml#id(w_238)"/>
</alt>
...
```

(corresponding DTD: altern.dtd)

I.e., for each element for which deviating information can be provided an `<alt>` element is listed and only that information that differs, is specified (here marked bold). Each of the `<alt>` elements contains an `<orig>` child element which has an href to the element the alternative information is specified for. Searching for alternatives would require to specify the element name of the elements for which alternative information could have been provided for and the name of the alternative elements (e.g., `alt`).

Corrections to be noted relative to a document can be realized along the same principles as alternative markup is described.

In addition one might wish to add the following information in order to encode the source for possible alternative descriptions (attribute `lack`):

- The annotator knows that the phenomenon must be encoded as either A, B, or C but cannot decide between them because of a lack of knowledge or decision criteria (`"lack=info"`).
- The theory used for the annotation is not as fine grained as to allow the distinction of phenomena in the data (`"lack=theory"`).
- There are a number of annotators which have labelled the data differently (`lack="agree"`).

### 3.3.2 Resource annotation

In general one should distinguish between two different kinds of representation encoded in XML:

- event stream tagging
- knowledge representation (or concept representation).

The first case - *event stream tagging* - (ideally) is the encoding of information provided for event phenomena (tokens) in context. The necessity for annotating this kind of information obviously stems from the fact that part of the information tagged cannot be predicted or calculated, i.e. there is no model for its determination. Thus, by annotating this information in corpora, a data collection that can be used for the development and evaluation of hypothesis is created. The second case - *knowledge representation* - is a collection of (linguistic) phenomena (types) and all information that is context independent or for the determination of which rules can be provided. The distinction made here presupposes that the aim of the research community is to enlarge the amount of information that can be stated and stored in knowledge representation databases. Event stream tagging is one means to approach to this goal.

Relations between lexical items may be quite complex and simple `href` attributes will be insufficient as most relations will exist between elements that are not neighbours. The following example illustrates how a thesaurus could be represented. First there is a file containing basic lexical elements.

lex.xml

```
...
<lem id="lem_002030" stem="animal"/>
<lem id="lem 020404" stem="bird"/>
```

```
<lem id="lem_100607" stem="cuckoo"/>
<lem id="lem_605003" stem="canary"/>
<lem id="lem_040508" stem="horse"/>
<lem id="lem_800300" stem="pony"/>
<lem id="lem_020803" stem="fish"/>
<lem id="lem_040000" stem="shark"/>
<lem id="lem_070308" stem="ale"/>
<lem id="lem_098030" stem="bright"/>
<lem id="lem_300830" stem="dark"/>
<lem id="lem_090800" stem="light"/>
...
```

(corresponding DTD: lex.dtd)

Now, there are two reasons why a thesaurus represented in XML should contain relation elements which include the lexical elements that are part of that relation: First, XML does not allow href attributes to contain pairs or tuples of elements which are not direct neighbours. Secondly, it would be very difficult to find a handy way of specifying the roles of the elements in a given relation. If this information were to be put into attributes of one element, then one would need an attribute that contains *n* substrings each of which specify the role of the elements included in that relation. This is not supported in XML or by XML parsers, thus the substrings of the roles and the ref attribute in

theswrong.xml

```
...
<theswrong id="theswrong_023" roles="role1 role2 role3" href="lex.xml#id(020404)..id(605003)"/>
...
```

(corresponding DTD: theswrong.dtd)

would neither be split into single items nor would there be a mechanism of checking if they contain the same number of items, nor would there be a mechanism of relating the first substring of the `roles` attribute to the first substring of the `ref` attribute, etc.

In the case of the following example

thesgood.xml

```
...
<thesgood id="thesgood_073" >
  <theselm id="theselm_045" role="role1" href="lex.xml#id(020404)"/>
  <theselm id="theselm_046" role="role2" href="lex.xml#id(100607)"/>
  <theselm id="theselm_047" role="role3" href="lex.xml#id(605003)"/>
</thesgood>
...
```

(corresponding DTD: thesgood.dtd)

it is easy to check whether there is the correct number of role attributes and it is clear which role specification belongs to which element.

It should also be obvious, that it would not be possible to specify the semantical relations inside the <lem> elements of the file lex.xml as any element (i.e. word) can have different semantical roles depending on the word counterpart it is compared to.

The following is an example of how thesaurus relations could be represented.

thes.xml

```
...
<thesrel type="syn">
  <theselm role="synonym" href="lex.xml#id(lem_090800)"/>
  <theselm role="synonym" href="lex.xml#id(lem_098030)"/>
</thesrel>
```

```
<thesrel type="isa">
  <theselm role="broadterm"  href="lex.xml#id(lem_002030)"/>
  <theselm role="narrowterm" href="lex.xml#id(lem_040508)"/>
</thesrel>
<thesrel type="isa">
  <theselm role="broadterm"  href="lex.xml#id(lem_040508)"/>
  <theselm role="narrowterm" href="lex.xml#id(lem_800300)"/>
</thesrel>
<thesrel type="ant">
  <theselm role="antonym" href="lex.xml#id(lem_3008302)"/>
  <theselm role="antonym" href="lex.xml#id(lem_0908008)"/>
</thesrel>
...
```

(corresponding DTD: thes.dtd)

A possible application for lexical information files for the process of annotation is the following: It might be the case that part-of-speech (POS) information for most words can be predicted by only looking up in a lexicon, thus it seems a waste to specify POS in every case, i.e. for every word regardless if this information is predictable or not. Thus, it might be useful to put all information predictable into a lexicon file and only specify the information in a to-be-tagged event stream document if it cannot be predicted.

Thus, one would have a lexicon file

lex2.xml

```
...
<lem id="lem_012121" pos="noun" num="sg" pron="tri:"  href="lex.xml#id(lem_014435)">tree</w>
<lem id="lem_012122" pos="noun" num="pl" pron="tri:z" href="lex.xml#id(lem_014435)">trees</w>
<lem id="lem_212652" pos="verb"          pron="gIv"   href="lex.xml#id(lem_009233)">give</w>
<lem id="lem_212653" pos="verb" num="sg" pron="gIvz"  href="lex.xml#id(lem_009233)">gives</w>
<lem id="lem_352678" pos="det"           pron="D@"    href="lex.xml#id(lem_013228)">the</w>
...
```

(corresponding DTD: lex2.dtd)

and a tag file that contains the utterance

*...they give the trees to...*

utter.xml

```
...
<w id="w_utt020" num="pl"            href="lex2.xml#id(lem_212652)"/>
<w id="w_utt021" num="pl" case="acc" href="lex2.xml#id(lem_012122)"/>
<w id="w_utt022"          case="acc" href="lex2.xml#id(lem_212652)"/>
...
```

(corresponding DTD: utter.dtd)

So in the actual markup, only the kind of information that cannot be determined by the form is specified. This is cost effective and easier to maintain, because once there is more information available for individual words, this information can be specified in the lexicon file and is accessible by the annotation element in the referring document. Also, local information, i.e. information specified in the elements in the utterance tag document overrides information specified in the lexicon document.

### 3.3.3 Annotation process

The last category of applications described here is not inspired by linguistics as such but is meant to provide markup that can be used to document the process of annotation itself.

### 3.3.3.1 Definitions

For a graphical user interface of other purposes, there might be the need of having access to definitions for elements, attributes and values available in XML documents. In the following paragraphs, means for the encoding of such information is provided.

Suppose, there is a sentence marked up in this way:

sentword.xml

```
...
<sent id="s01" type="ass" who="mary">
  <word id="w01" pos="ART" num="sg">The</word>
  <word id="w02" pos="NN"  num="sg">boy</word>
  <word id="w03" pos="V"   num="sg">saw</word>
  <word id="w04" pos="ART" num="pl">the</word>
  <word id="w05" pos="NN"  num="pl">stars</word>
</sent>
...
```

(corresponding DTD: sentword.dtd)

And one central file is used to provide definitions for the element type strings, attribute type strings, and value strings used in the document. Then its structure could look like this:

def.xml

```
...
<defs>
  <element id="e01"     name="sent">Sentences are made up of words.
    <attribute id="a01" name="type">The type of sentence is described by type.
      <value id="v01"   name="ass">An assertion</value>
      <value id="v02"   name="quest">A question</value>
    </attribute>
    <attribute id="a02" name="who">Speakers are defined by the who attribute.
      <value id="v03"   name="mary">Mary B.</value>
      <value id="v04"   name="peter">Peter A.</value>
    </attribute>
  </element>
  <element id="e02"     name="word">Words are all things in between white space.
    <attribute id="a03" name="pos">Part of speech is defined in the pos attribute.
      <value id="v05"   name="ART">article</value>
      <value id="v06"   name="NN">noun</value>
      <value id="v07"   name="V">verb</value>
    </attribute>
    <attribute id="a04" name="num">The number is defined in the num attribute.
      <value id="v08"   name="sg">singular</value>
      <value id="v09"   name="pl">plural</value>
    </attribute>
  </element>
</defs>
...
```

(corresponding DTD: def.dtd)

Now, how can we make links between markup information (elements, attributes, values) and their definitions? It is obvious that this linking information should be provided automatically, i.e. the coder will not want to specify links to definition elements every time s/he specifies a new value for a given element. So, after the annotation process a mechanism would provide links between the locations where elements/attributes/values are used and their definition element. This could be done by queries. After that there would be a document providing the linking infrastructure.

deflink.xml

```
...
<deflink href="def.xml#id(e01)">
  <used href="sentword.xml#id(s01)"/>
</deflink>
<deflink href="def.xml#id(e02)">
  <used href="sentword.xml#id(w01)"/>
  <used href="sentword.xml#id(w02)"/>
```

```
<deflink href="def.xml#id(v01)">
  <used href="sentword.xml#id(s01)"/>
</deflink>
<deflink href="def.xml#id(v02)">
</deflink>
<deflink href="def.xml#id(v03)">
```

```
  <used href="sentword.xml#id(w03)"/>
  <used href="sentword.xml#id(w04)"/>
  <used href="sentword.xml#id(w05)"/>
</deflink>
<deflink href="def.xml#id(a01)">
  <used href="sentword.xml#id(s01)"/>
</deflink>
<deflink href="def.xml#id(a02)">
  <used href="sentword.xml#id(s01)"/>
</deflink>
<deflink href="def.xml#id(a03)">
  <used href="sentword.xml#id(w01)"/>
  <used href="sentword.xml#id(w02)"/>
  <used href="sentword.xml#id(w03)"/>
  <used href="sentword.xml#id(w04)"/>
  <used href="sentword.xml#id(w05)"/>
</deflink>
<deflink href="def.xml#id(a04)">
  <used href="sentword.xml#id(w01)"/>
  <used href="sentword.xml#id(w02)"/>
  <used href="sentword.xml#id(w03)"/>
  <used href="sentword.xml#id(w04)"/>
  <used href="sentword.xml#id(w05)"/>
</deflink>
```

```
  <used href="sentword.xml#id(s01)"/>
</deflink>
<deflink href="def.xml#id(v04)">
</deflink>
<deflink href="def.xml#id(v05)">
  <used href="sentword.xml#id(w01)"/>
  <used href="sentword.xml#id(w04)"/>
</deflink>
<deflink href="def.xml#id(v06)">
  <used href="sentword.xml#id(w02)"/>
  <used href="sentword.xml#id(w05)"/>
</deflink>
<deflink href="def.xml#id(v07)">
  <used href="sentword.xml#id(w03)"/>
</deflink>
<deflink href="def.xml#id(v08)">
  <used href="sentword.xml#id(w01)"/>
  <used href="sentword.xml#id(w02)"/>
  <used href="sentword.xml#id(w03)"/>
</deflink>
<deflink href="def.xml#id(v09)">
  <used href="sentword.xml#id(w04)"/>
  <used href="sentword.xml#id(w05)"/>
</deflink>
...
```

(corresponding DTD: deflink.dtd)

I.e. for each definition element there is a list of elements where its corresponding entity (element, attribute, value) has been used.

One could use a similar mechanism for identifying element-names, attributes or values which have no definition or element-types, attributes, or values which are never used.

### 3.3.3.2 Comments

Any entity that is specified during the annotation process may require commenting, the selection of a section and its definition as a given element type or the choice of a particular attribute value may need a comment. Thus, there should be an extra element for comments. If the markup for which comments are necessary is the following

sentword.xml

```
...
<sent id="s01" type="ass" who="mary">
  <word id="w01" pos="ART" num="sg">The</word>
  <word id="w02" pos="NN"  num="sg">boy</word>
  <word id="w03" pos="V"   num="sg">saw</word>
  <word id="w04" pos="ART" num="pl">the</word>
  <word id="w05" pos="NN"  num="pl">stars</word>
</sent>
...
```

(corresponding DTD: sentword.dtd)

a comment file could look like this:

comment.xml

```
...
<cmnt id="cmnt_001" href="sent.xml#id(s01)">
  Not quite clear if the segmentation is correct, there could also be a pause.
</cmnt>
<cmnt id="cmnt_001" att="num" href="sent.xml#id(w03)">
  The number can only be specified by looking at the sentence context.
</cmnt>
...
```

(corresponding DTD: comment.dtd)

The idea is to have comments with an href to the elements that are to be commented and a PCDATA child that is the comment. If an attribute value is commented, then there is a special attribute att the value of which is the string which is used for the attribute in the element referred to.

### 3.3.3.3 Annotation history

Marking up the history of elements in this environment requires that consecutive versions of elements or level codings as such be kept in different files but that the elements keep their id's across different versions of the level markup because of elements from other files referring to these elements. Thus, for each element, there would be a markup for its changes. No markup is suggested here, that keeps track of changes on the level of characters during an annotation session.

Consider the following files:

word001.xml (version from 23 Aug 1997)

```
...
<word id="word_001" pos="art">the</word>
<word id="word_002" pos="nn">house</word>
<word id="word_003" pos="v">was</word>
<word id="word_004" pos="adj">new</word>
...
```

(corresponding DTD: words.dtd)

word002.xml (version from 23 Aug 1998)

```
...
<word id="word_001"  pos="defart">the</word>
<word id="word_002"  pos="nnsg">house</word>
<word id="word_003"  pos="v">was</word>
<word id="word_003a" pos="adv">still</word>
<word id="word_004"  pos="adj">new</word>
...
```

(corresponding DTD: words.dtd)

Then a history markup could look the following way:

hist.xml

```
...
<hist id="hist_001">
  <histelm id="histelm_001" href="word001.xml#id(word_001)"/>
  <histelm id="histelm_002" href="word002.xml#id(word_001)"/>
</hist>
<hist id="hist_002">
  <histelm id="histelm_003" href="word001.xml#id(word_002)"/>
  <histelm id="histelm_004" href="word002.xml#id(word_002)"/>
</hist>
<hist id="hist_003">
  <histelm id="histelm_005" href="word001.xml#id(word_003)"/>
  <histelm id="histelm_006" href="word002.xml#id(word_003)"/>
</hist>
<hist id="hist_004">
  <histelm id="histelm_007" href="word001.xml#id(word_004)"/>
  <histelm id="histelm_008" href="word002.xml#id(word_004)"/>
</hist>
...
```

(corresponding DTD: hist.dtd)

So there is a history element for all those `<word>` element tuples that have identical id's (the identification of these is possible with a query).

#### 3.3.3.4 Resource file

For a project it is desirable to know which files belong to that project and keep this information. One implicit way of keeping information of files belonging to the same project together (in XML) are href values whenever the markup of one type is referring to existing markup in another file. However, this way of obtaining information cannot guarantee completeness and requires knowledge about the top level element file. Thus, the creation and maintenance of a general resource file is indispensable. In order to support the automatic creation and maintenance of such a resource file, there must be a way of detecting what files created belong to what project. Thus, in addition to some general attributes (more below) an attribute `project` is needed for every XML file that states the group membership of this file. When assigning a `project` name to a (newly created) XML file, an equivalent resource file will be updated adding the file name of that XML file to the list. In the following paragraphs, the element names, their attributes and values for a resource file are described.

**`<project>`**

The `<project>` element is a container of a list of <file> elements.

The following two attributes are recommended to be used with `<project>` elements.

**`id`**

The `id` attribute is required. Its value must be unique within a document.

**`prjname`**

The `prjname` attribute is required. It states the name of the project and enables the identification of all files that belong to a project.

> **`<file>`**
>
> The `<file>` element is an element that keeps minimal information on a file that has been produced during the project. It may be a binary file, like a sound file or a video file, an XML file or just any other ASCII (non-binary) file. For each new file produced, one corresponding `<file>` element is created in a resource file. `<file>` elements require existing files but do not refer to elements within these like in other cases where `href` attributes are used to make links between existing elements and new elements.
>
> The following attributes are recommended to be used with `<file>` elements.
>
> **`id`**
>
> The `id` attribute is required. Its value must be unique within a document.
>
> **`fenc`**

The `fenc` attribute is required. The `fenc` attribute indicates the representation format of the information of the file and is needed in order to determine the kind of processing of the data. Distinctions are made between binary files [bin], XML files [xml] and other ASCII files [asc]. More distinctions may be needed in order to control the behaviour of software interpreting and manipulating the files. Possible values are `[bin, xml, asc]`.

**`fname`**

The `fname` attribute is required. The `fname` attribute indicates the file name of the file listed.

***Example***

As example consider that there are three files: A speech file (*mary.au*), a lexicon file in XML (*lex.xml*), a non-XML transcription of the text (*word.txt*), a XML representation of the word level (*word.xml),* and a coreference annotation file (*coref.xml*). They all belong to the project called *Mary's first words*.

In the corresponding resource file (*reso.xml*), one would have three entries.

reso.xml

```
<project id="proj_001" prjname="Mary's first words">
   <file id="file_002" fenc="bin" fname="mary.au"/>
   <file id="file_003" fenc="xml" fname="lex.xml"/>
   <file id="file_004" fenc="asc" fname="word.txt"/>
   <file id="file_005" fenc="xml" fname="word.xml"/>
   <file id="file_006" fenc="xml" fname="coref.xml"/>
   <file id="file_007" fenc="asc" fname="word.txt"/>
</project>
```

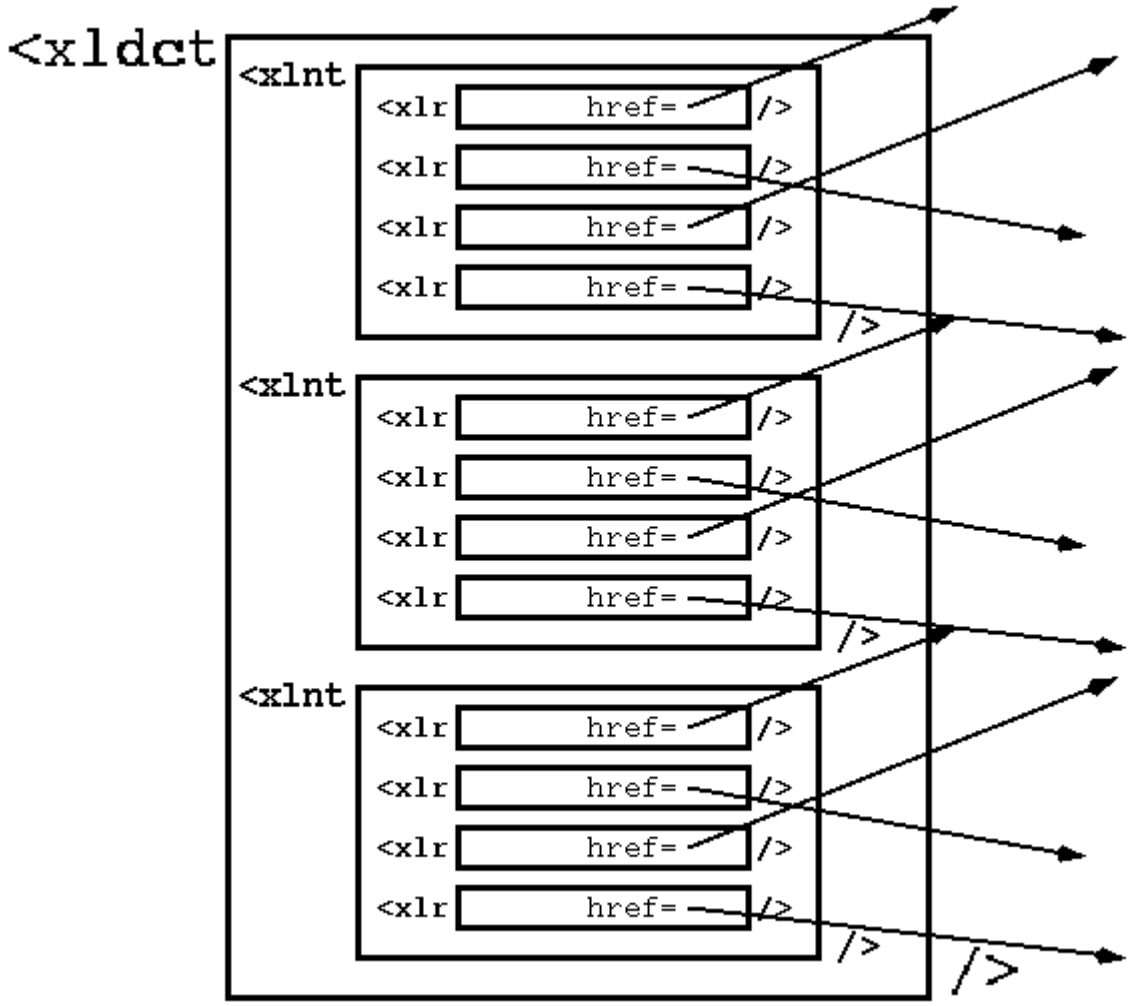
(corresponding DTD: reso.dtd)

# 4 General cross-level annotation

In this section general cross-level markup entities and properties are described. As specified before, these are conceptualized as basic tools for ad hoc and preliminary use when existing XML elements are considered being the basis of further markup. The structure and names defined hereafter can be changed and be adopted for special purposes. Most aspects of this general markup are designed for maximal processibility of the XML structure.

## 4.1 Markup Declaration

There are three kinds of elements (`<xldct>`, `<xlnt>`, and `<xlr>`) which are described below. The `<xldct>` elements contain a set of `<xlnt>` elements which contain `<xlr>` elements. The idea of this structure is, that there is one document for a list of phenomena of a given type. This whole document is held inside the `<xldct>` element. The individual phenomena are addressed by `<xlnt>` elements. Each of them contains a tuple of `<xlr>` elements each of which points to elements from other documents (see figure).




This set of elements offers what is needed to represent phenomena whose nature is defined by the relation of predefined phenomena and is also conformant with the layout designed for query results in the Q4M processor. Attributes and values of the cross-level markup defined hereafter focus on description needed for further processing and description of formal aspects of the markup structure. Additional attributes specifying theoretical description of phenomena may be added according to the application of the markup to individual phenomena that are marked up with this markup.

## 4.2 Description of Elements

In the following paragraphs, the elements and their attributes are described in detail.

### 4.2.1 `<xlr>`

***Description***

The `<xlr>` (cross-level reference) element is an element the most important function of which is to refer (by an href attribute) to an element of another existing document. This is its most important function. The technique of using embedded linking elements has been introduced in section *3.2 Principles.* The need to do so lies in the fact that syntactically it is not allowed to use

values in the href attribute which are no direct neighbours. The functional need to have separate pointing elements is that this structure allows for a more powerful representation of additional aspects of the roles of the elements pointed to in this cross-level relation.

### *Data Source*

<xlr> elements require existing XML elements (in XML documents) with a DTD that define attributes of type ID as required and which are valid XML documents (cf. the example).

minimaldoc.xml

```
<?xml version="1.0"?>
<!DOCTYPE doc [
<!ELEMENT minimelm (#PCDATA)>
<!ATTLIST minimelm
          id       ID #REQUIRED>
]>
<doc>
  <minimelm id="minimelm_01"/>
</doc>
```

Without the id attribute, it is impossible to link to the existing elements with href attributes.

### *Segmentation/Selection*

Only individual elements from the existing document(s) can be selected as being pointed to by href attributes of <xlr> elements. There are no other formal criteria for the selection process as the annotation process may be done by hand and is intended as a collection process only or the process may be automated and formally specified by a query expression.

### *Assignment*

The following attributes are recommended to be used with <xlr> elements.

**id**

The id attribute is required. Its value must be unique within a document.

**href**

This is a reference to another element, it is required. The exact syntax can be found in [5].

**who**

This element states the identity of the producer of the markup. This may be a software module or a human annotator. Preferably, the strings allowed for the who values are a fixed set of strings specified in the DTD. This attribute is optional.

**cmt**

This is a comment that can include any information the annotator or anybody else needs. The cmt value is a string. This attribute is optional.

**cert**

This is an information about the certainty value of the appropriateness of the selection of the element pointed to. If the selection of the element has been made by a software module, the value will be 1. If the selection has been made by human annotation, the value may be lower and will reflect the lack of criteria for an operationalisation of the selection process. Possible values are `[0 .1 .2 .3 .4 .5 .6 .7 .8 .9 1]`. This attribute is optional.

**`stat`**

This is the status of the `<xlr>` element. The value of the `stat` attribute reflects the number of review steps that have been applied or will have to be applied. In the case where a query processor has provided the `<xlr>` elements, the value of this attribute will be `done`. Possible values are: `[draft, reviewed, done]`. This attribute is optional.

**`elmname`**

This is information specifies the type of element that is referred to as this information cannot be retrieved unless the `href` attribute value is resolved. The values of this attribute are strings. This attribute is optional.

**`restyp`**

This information specifies if the document that holds the element referred to is an event stream type of document or a knowledge resource document. This information is helpful in case the reliability of the information of the elements pointed to by different `<xlr>` elements is to be evaluated and adopted from one to another. Corresponding values are `[evs, knr]`. This attribute is optional.

**`refvar`**

This attribute specifies the variable name that was used as reference to the element in a (Q4M) query expression if the document is a result of a query process and mandatory for queries that reuse existing query output for further refinement. Possible values are strings. This attribute is optional.

**`props`**

This attribute is required. It contains information on those properties of the element (that is referred to by the `href` attribute) that are constitutive for its selection. Ideally, this is a list of query sub-expressions which contain reference to the element. In the case of manual assignment at least a list of the attributes is needed. The values of this attribute are strings.

### *Example*

As example consider that one wants to study repetitions of words in utterances such as *"The ball...ball was red*". For this task, a speech file segmented on word level is available. Two more aspect shall be important for this task: The collection of data is restricted to nouns. Therefore, the inclusion of lexical resources is indispensable. In addition, only those nouns are taken into account, which have a frequency of 500 or more in the corpus that was used to produce the lexical resource.

Thus, there is a resource file which lists individual word entries (lexw.xml), a resource file which lists lemma entries (lexf.xml) and a file which is a word transcription of a given conversation (wordutt.xml).

lexw.xml

```
...
<lexw id="lexw_00766" freq="433"  href="lexf.xml#id(lexf_0487)">
  cow
</lexw>
<lexw id="lexw_00767" freq="2121" href="lexf.xml#id(lexf_0487)">
  cows
</lexw>
<lexw id="lexw_01221" freq="1212" href="lexf.xml#id(lexf_2121)">
  house
</lexw>
<lexw id="lexw_01222" freq="4121" href="lexf.xml#id(lexf_2121)">
  houses
</lexw>
<lexw id="lexw_20235" freq="33"   href="lexf.xml#id(lexf_3768)">
  wall
</lexw>
<lexw id="lexw_20236" freq="733"  href="lexf.xml#id(lexf_3768)">
  walls
</lexw>
...
```

(corresponding DTD: lexw.dtd)

lexf.xml

```
...
<lexf id="lexf_0487" pos="noun" base="cow"/>
<lexf id="lexf_2121" pos="noun" base="house"/>
<lexf id="lexf_3768" pos="noun" base="wall"/>
...
```

(corresponding DTD: lexf.dtd)

wordutt.xml

```
...
<w id="w_0232">They</w>
<w id="w_0233">built</w>
<w id="w_0234">the</w>
<w id="w_0235">houses</w>
<w id="w_0236">houses</w>
<w id="w_0237">and</w>
<w id="w_0238">left</w>
...
<w id="w_4409">And</w>
<w id="w_4410">there</w>
<w id="w_4411">was</w>
<w id="w_4412">a</w>
<w id="w_4413">cow</w>
<w id="w_4414">cow</w>
...
<w id="w_5098">She</w>
<w id="w_5199">painted</w>
<w id="w_5100">the</w>
<w id="w_5101">wall</w>
<w id="w_5102">wall</w>
<w id="w_5103">red</w>
...
```

(corresponding DTD: wordutt.dtd)

In this case for each individual conjunction of this type, one would need four <xlr> elements:

- one pointing to the first <w> occurance of the word
- one pointing to the second <w> occurance of the word
- one pointing to the <lexw> the words belong to
- one pointing to <lexf> element the words belong to

A compatible query in Q4M would be:

```
($w1 w)($w2 w)($lw lexw)($ll: lexf); ($w1 <>1 $w2) and ($w1 # ~ $w2 #) and
($w1 # ~ $lw #) and ($lw freq >= 500) and ($lw 1^ $ll) and ($ll pos ~
"noun")
```

Thus one would have the following <xlr> elements (for reasons of clarity, the layout has been changed).

xlr.xml

```
...
<xlr
 id="xlr_001"
 href="wordutt.xml#id(w_0235)"
 who="Q4M"
 cmt="first word"
 cert="1"
 stat="done"
 elmname="w"
 restyp="evs"
 refvar="w1"
 props="($d 1^ $w1) and
        ($w1 <>1 $w2) and
```

```
...
<xlr
 id="xlr_052"
 href="wordutt.xml#id(w_4413)"
 who="Q4M"
 cmt="first word"
 cert="1"
 stat="done"
 elmname="w"
 restyp="evs"
 refvar="w1"
 props="($d 1^ $w1) and
        ($w1 <>1 $w2) and
```

```
...
<xlr
 id="xlr_127"
 href="wordutt.xml#id(w_5101)"
 who="Q4M"
 cmt="first word"
 cert="1"
 stat="done"
 elmname="w"
 restyp="evs"
 refvar="w1"
 props="($d 1^ $w1) and
        ($w1 <>1 $w2) and
```

```
       ($w1 # ~ $w2 #) and
       ($w1 # ~ $lw #)"
/>
<xlr
 id="xlr_002"
 href="wordutt.xml#id(w_0236)"
 who="Q4M"
 cmt="second word"
 cert="1"
 stat="done"
 elmname="w"
 restyp="evs"
 refvar="w2"
 props="($d 1^ $w2) and
       ($w1 <>1 $w2) and
       ($w1 # ~ $w2 #)"
/>
<xlr
 id="xlr_003"
 href="lexw.xml#id(lexw_01222)"
 who="Q4M"
 cmt="word type"
 cert="1"
 stat="done"
 elmname="lexword"
 restyp="knr"
 refvar="lw"
 props="($lw freq >= 500) and
       ($lw 1^ $ll)"
/>
<xlr
 id="xlr_004"
 href="lexf.xml#id(lexf_2121)"
 who="Q4M"
 cmt="word lemma"
 cert="1"
 stat="done"
 elmname="lexlem"
 restyp="knr"
 refvar="ll"
 props="($lw 1^ $ll) and
       ($ll pos ~
        &amp;quot;noun&amp;quot;)"
/>
...
```

```
       ($w1 # ~ $w2 #) and
       ($w1 # ~ $lw #)"
/>
<xlr
 id="xlr_053"
 href="wordutt.xml#id(w_4414)"
 who="Q4M"
 cmt="second word"
 cert="1"
 stat="done"
 elmname="w"
 restyp="evs"
 refvar="w2"
 props="($d 1^ $w2) and
       ($w1 <>1 $w2) and
       ($w1 # ~ $w2 #)"
/>
<xlr
 id="xlr_054"
 href="lexw.xml#id(lexw_00766)"
 who="Q4M"
 cmt="word type"
 cert="1"
 stat="done"
 elmname="lexword"
 restyp="knr"
 refvar="lw"
 props="($lw freq >= 500) and
       ($lw 1^ $ll)"
/>
<xlr
 id="xlr_055"
 href="lexf.xml#id(lexf_0487)"
 who="Q4M"
 cmt="word lemma"
 cert="1"
 stat="done"
 elmname="lexlem"
 restyp="knr"
 refvar="ll"
 props="($lw 1^ $ll) and
       ($ll pos ~
        &amp;quot;noun&amp;quot;)"
/>
...
```

```
       ($w1 # ~ $w2 #) and
       ($w1 # ~ $lw #)"
/>
<xlr
 id="xlr_128"
 href="wordutt.xml#id(w_5102)"
 who="Q4M"
 cmt="second word"
 cert="1"
 stat="done"
 elmname="w"
 restyp="evs"
 refvar="w2"
 props="($d 1^ $w2) and
       ($w1 <>1 $w2) and
       ($w1 # ~ $w2 #)"
/>
<xlr
 id="xlr_129"
 href="lexw.xml#id(lexw_20235)"
 who="Q4M"
 cmt="word type"
 cert="1"
 stat="done"
 elmname="lexword"
 restyp="knr"
 refvar="lw"
 props="($lw freq >= 500) and
       ($lw 1^ $ll)"
/>
<xlr
 id="xlr_130"
 href="lexf.xml#id(lexf_3768)"
 who="Q4M"
 cmt="word lemma"
 cert="1"
 stat="done"
 elmname="lexlem"
 restyp="knr"
 refvar="ll"
 props="($lw 1^ $ll) &amp;&amp;
       ($ll pos ~
        quot;noun&amp;quot;)"
/>
...
```

(corresponding DTD: xlr.dtd)

### *Coding Procedure*

If no software for the automated processing is available, the following procedure can be used:

| **procedural guideline** |
| --- |
| • formulate the constraints/the description of the phenomenon in terms of information available. This includes<br>  • knowledge on the elements available<br>  • knowledge on the attributes and values of the elements available in the data or procedures for the generation of the necessary information on properties<br>  • knowledge on the exact constellation of the elements and their properties<br>  • the elements involved<br>  • their properties and their relations to one another<br>  • the combination of individual properties or |

<table>
<tr><td colspan="2">relations of elements</td></tr>
<tr><td colspan="2">Then, there are two options: A standard procedure and an optimized procedure. The optimized procedure aims at fast results.</td></tr>
<tr><th>standard procedure</th><th>optimized procedure</th></tr>
<tr><td>
<ul>
<li>for each element type defined
<ul>
<li>look for tokens of these elements</li>
<li>check whether they have the properties necessary</li>
<li>check whether they are in the relations to other elements specified</li>
<li>markup those sets of elements that match all the requirements</li>
</ul>
</li>
</ul>
</td><td>
<ul>
<li>for each of the value constraints defined
<ul>
<li>begin from the type of element which are fewest
<ul>
<li>for each of the elements found, check the value constraints in an order that starts with those properties which are least likely fulfilled, delete elements from the list whenever they do not match a criterion</li>
</ul>
</li>
<li>continue with the next least frequent type of elements until all value constraints of the elements have been checked</li>
</ul>
</li>
<li>for all of the constraints involving two elements
<ul>
<li>check the constraints in an order that starts with those which are least likely fulfilled and keep only those elements which match all of the relation criteria</li>
</ul>
</li>
<li>for all of the logical constraints
<ul>
<li>again check the constraints in an order that starts with those which are least likely fulfilled and keep only those element pairs for which all of the logical constraints are true</li>
</ul>
</li>
<li>markup those sets of elements that match the requirements</li>
</ul>
This procedure aims at identifying elements that do not match as early as possible in order to reduce the number of checks.
</td></tr>
</table>

### *Markup Table*

The following table summarizes the attributes and their values.

| **Attributes and values of `<xlr>` elements** | |
|---|---|
| **attribute** | **values** |
| `id` | `[ASCII]` |
| `href` | `[ASCII]` |

| | |
|---|---|
| who | [ASCII] |
| cmt | [ASCII] |
| cert | 0 .1 .2 .3 .4 .5 .6 .7 .8 .9 1 |
| stat | draft, reviewed, done |
| elmname | [ASCII] |
| restyp | evs, knr |
| refvar | [ASCII] |
| props | [ASCII] |

### 4.2.2 <xlnt>

***Description***

The <xlnt> (cross-level element) element is the container for the concept that includes a tuple of <xlr> elements which point to a collection of elements. The number of included <xlr> elements will in part depend on the application, and if the document is the output of a query, any number is possible. Typical numbers of sub <xlr> elements and associated applications - other than queries - will be:

- **one element**: alternatives, corrections,
- **two elements**: thesauri (antonyms), translation dictionaries
- **more elements**: thesauri (synonyms, hyponymy)

***Data Source***

<xlnt> elements require existing <xlr> elements. These can be placed in another document, but this is not recommended. In that case, an additional href attribute is needed.

***Segmentation/Selection***

For each <xlnt> element corresponding to a token of a special phenomenon identified by inspecting existing markup elements, a set of <xlr> elements is grouped together as children elements of this <xlnt> element.

***Assignment***

The following attributes are recommended to be used with <xlr> elements.

**id**

The id attribute is required. Its value must be unique within a document.

**href**

This is a reference to a tuple of <xlr> elements in case they are not embedded inside the <xlnt> element but reside in another file.

**who**

This element states the identity of the producer of the markup. This may be a software module or a human annotator. Preferably, the strings allowed for the `who` values are a fixed set of strings specified in the DTD. This attribute is optional.

**`cmt`**

This is a comment that can include any information the annotator or anybody else needs. The cmt value is a string. This attribute is optional.

**`desc`**

This is a description of the phenomenon.

**`phename`**

This information contains a name to be used for the phenomenon that is addressed by the `<xlnt>` elements. This is a string value. This attribute is optional.

**`cert`**

This is an information about the certainty value of the appropriateness of the selection of the element pointed to. If the selection of the element has been made by a software module, the value will be 1. If the selection has been made by human annotation, the value may be lower and will reflect the lack of criteria for an operationalisation of the selection process. Possible values are `[0 .1 .2 .3 .4 .5 .6 .7 .8 .9 1]`. This attribute is optional.

**`stat`**

This is the status of the `<xlr>` element. The value of the `stat` attribute reflects the number of review steps that have been applied or will have to be applied. In the case where a query processor has provided the `<xlr>` elements, the value of this attribute will be `done`. Possible values are: `[draft, reviewed, done]`. This attribute is optional.

**`calc`**

This attribute contains an expression which specifies a procedure for the calculation of an attribute value by using values of the elements included by the `<xlr>` elements. The syntax of the value is not defined but may be an extension of the MSL stylesheet expression syntax. This attribute is optional.

### *Example*

As example consider the `<xlr>` element document produced above (xlr.xml). For the markup of `<xlnt>` elements, each tuple of `<xlr>` elements is put inside an `<xlnt>` element.

xlnt.xml

```
...
<xlnt
 id="xlnt_001"
 href="xlr.xml#id(xlr_001)..id(xlr_004)"
 who="Q4M"
 cmt="why cows?"
 desc="repitions of nouns caused by high token frequency"
 phename="rnhf"
 cert="1"
 stat="done"
```

```
 calc="w.freq"
/>
...
<xlnt
 id="xlnt_007"
 href="xlr.xml#id(xlr_052)..id(xlr_055)"
 who="Q4M"
 cmt="houses"
 desc="repitions of nouns caused by high token frequency"
 phename="rnhf"
 cert="1"
 stat="done"
 calc="w.freq"
/>
...
<xlnt
 id="xlnt_021"
 href="xlr.xml#id(xlr_127)..id(xlr_130)"
 who="Q4M"
 cmt="wall"
 desc="repitions of nouns caused by high token frequency"
 phename="rnhf"
 cert="1"
 stat="done"
 calc="w.freq"
/>
...
```

(corresponding DTD: xlnt.dtd)

### *Coding Procedure*

Each group of <xlr> is referred to by href attributes of <xlnt> elements or embedded inside the <xlnt> element. Whichever is easier should be done. This is mainly a syntactical operation.

### *Markup Table*

The following table summarizes the attributes and their values.

| Attributes and values of <xlnt> elements | |
|---|---|
| **attribute** | **values** |
| id | [ASCII] |
| who | [ASCII] |
| cmt | [ASCII] |
| desc | [ASCII] |
| phename | [ASCII] |
| cert | 0 .1 .2 .3 .4 .5 .6 .7 .8 .9 1 |
| stat | draft, reviewed, done |
| calc | [ASCII] |

## 4.2.3 <xldct>

### *Description*

The <xldct> (cross-level document) element is the container of the cross-level elements (<xlnt>). The <xldct> element is perceived as a list element the entries of which are <xlnt> elements each of which contains a set of pointers held by <xlr> elements that ensemble are necessary to define examples of the concept addressed.

### *Data Source*

<xldct> elements require existing <xlnt> elements. These can be placed in another document, but this is not recommended. In that case, an additional href attribute is needed.

### *Segmentation/Selection*

As the <xldct> element is a set element in that it contains all those <xlnt> elements that describe the extension (all tokens/occurences) of a special phenomenon, this set of <xlnt> elements is grouped together as children elements of this <xldct> element.

### *Assignment*

The following attributes are recommended to be used with <xlr> elements.

**id**

The id attribute is required. Its value must be unique within a document.

**href**

This is a reference to a tuple of <xlnt> elements in case they are not embedded inside the <xldct> element but reside in another file.

**who**

This element states the identity of the producer of the markup. This may be a software module or a human annotator. Preferably, the strings allowed for the who values are a fixed set of strings specified in the DTD. This attribute is optional.

**cmt**

This is a comment that can include any information the annotator or anybody else needs. The cmt value is a string. This attribute is optional.

**cert**

This is an information about the certainty value of the appropriateness of the selection of the element pointed to. If the selection of the element has been made by a software module, the value will be 1. If the selection has been made by human annotation, the value may be lower and will reflect the lack of criteria for an operationalisation of the selection process. Possible values are [0 .1 .2 .3 .4 .5 .6 .7 .8 .9 1]. This attribute is optional.

**stat**

This is the status of the <xlr> element. The value of the stat attribute reflects the number of review steps that have been applied or will have to be applied. In the case where a query processor has provided the <xlr> elements, the value of this attribute will be done. Possible values are: [draft, reviewed, done]. This attribute is optional.

**srtelm**

This is information specifying how the children elements (<xlnt>) are sorted. This attribute needs to be evaluated together with the srtatt, the srtcrt, and the srtord attributes. Attribute values are measurements or categorizations and thus fixations in our conceptual space. This

allows to compare them or to sort them according to any attribute. The standard order of elements within annotation documents are their hierarchical relationships (parent node before child node) and time (i.e. the sequence of occurrence). Since it is possible to extract or display selected types of elements, sorting criteria other than time/occurrance are possible and also have applications, e.g. duration, the part of speech value or the position number of the lemma entry they belong to in a lexicon.

Documents of utterances are 'more' static because they have a fixed order whereas the order in a lexicon is arbitrary and only relative to some formal aspects, they don't encode events (and categorization, description) but knowledge which is context independent and thus no special order of elements is required since the relative neighbourhood of elements does not provide any information. This does not necessarily mean that there is no use in sorting elements of lexical documents (i.e. for fast access of the most frequent entries). What is said here is that the order in a sequence of lexical elements does not correspond to any information important for the description of the nature of individual elements.

The `srtelm` attribute specifies the properties of which elements pointed to by `<xlr>` elements were exploited in order to define the order of `<xlnt>` elements. The elements can be specified best by the value of the respective `refvar` attribute. Less reliable is a reference by the `elname` value.

This attribute is optional.

**`srtatt`**

The `srtatt` attribute is only required and applicable if the `srtelm` attribute is specified. The `srtatt` element specifies which attribute of the element specified by the `srtelm` attribute is the relevant sorting criterion for the order of `<xlnt>` elements. This is a string value.

**`srtcrt`**

The `srtcrt` attribute is only required and applicable if the `srtatt` attribute is specified. The `srtcrt` element specifies if the sorting of values of the attributes specified by the `srtatt` attribute is done by string comparison or by a numerical evaluation.. Possible values are `[str, num]`.

**`srtord`**

The `srtord` attribute is only required and applicable if the `srtcrt` attribute is specified. The `srtord` element specifies the sorting direction of values ordered by a criterion defined by the `srtcrt` attribute. The direction can be normal or reverse. Possible values are `[std, rev]`.

**`crea`**

The `crea` attribute is required. This information specifies the algorithm used to select the tuples of elements under each `<xlnt>` element. In case a query processor was used to select the data, this will be a query expression. This can be free text or a query expression.

**`refnstab`**

This attribute specifies if the number of `<xlr>` elements within each `<xlnt>` element remains stable. Possible values are [`yes, no`]. This attribute is optional.

**`reftstab`**

This attribute specifies if the type of elements referred to by the <xlr> elements remains the same across different <xlnt> elements. Possible values are [yes, no]. This attribute is optional.

***Example***

As example consider the <xlnt> element document produced above (xlnt.xml). For the markup of <xldct> elements, all <xlnt> elements is put inside an <xlnt> element.

xldct.xml

```
...
<xldct
 id="xldct_001"
 href="xlnt.xml#id(xlnt_001)..id(xlr_033)"
 who="Q4M"
 cmt="are there semantic issues involved?"
 cert="1"
 stat="done"
 srtelm="lw"
 srtatt="freq"
 srtcrt="num"
 srtord="rev"
 crea="($w1 w)($w2 w)($lw lexw)($ll lexw);
       ($w1 <>1 $w2) and ($w1 # ~ $w2 #) and
       ($w1 # ~ $lw #) and ($lw freq >= 500) and
       ($lw 1^ $ll) and ($ll pos ~ &quot;noun&quot;)"
 refnstab="yes"
 reftstab="yes"
/>
...
```

(corresponding DTD: xldct.dtd)

***Coding Procedure***

Each group of <xlr> is referred to by href attributes of <xlnt> elements or embedded inside the <xlnt> element. Whichever is easier should be done. This is mainly a syntactical operation.

***Markup Table***

The following table summarizes the attributes and their values.

**Attributes and values of <xldct> elements**

| attribute | values |
|---|---|
| id | [ASCII] |
| who | [ASCII] |
| cmt | [ASCII] |
| cert | 0 .1 .2 .3 .4 .5 .6 .7 .8 .9 1 |
| stat | draft, reviewed, done |
| srtelm | [ASCII] |
| srtatt | [ASCII] |
| srtcrt | str, num |
| srtord | std, rev |
| crea | [ASCII] |
| refnstab | yes, no |
| reftstab | yes, no |

## References

[1] Klein, M., Bernsen, N.O., Davies, S., Dybkjær, L., Garrido, J., Kasch, H., Mengel, A., Pirrelli, V., Poesio, M., Quazza, S. and Soria, S.: Supported Coding Schemes. MATE Deliverable D1.1, July 1998. http://www.ims.uni-stuttgart.de/~mengel/papers/mateD11.html

[2] XML Linking Language (XLink), http://www.w3.org/TR/xlink

[3] Mengel, A. and Heid, U.: Enhancing Reusability of Speech Corpora by Hyperlinked Query Output. Eurospeech 99, Budapest, September 1999. http://www.ims.uni-stuttgart.de/~mengel/papers/eurospeech-budapest-99.ps.gz

[4] TEI Guidelines for Electronic Text Encoding, C.M. Sperberg-McQueen and Lou Burnard, editors, http://etext.lib.virginia.edu/TEI.html

[5] Isard, A., McKelvie, D. and Thompson, H.S.: Towards a Minimal Standard for Dialogue Transcripts: A New Sgml Architecture for the HCRC Map Task Corpus. *Proceedings of the 5th International Conference on Spoken Language Processing*, ICSLP98, Sydney. http://www.cogsci.ed.ac.uk/~dmck/Papers/icslp98.ps