STEVIN-IRME WP2, deliverable 2.1

Report on formalizing and elaborating the parameterized Equivalence Class Method for Dutch

Nicole Grégoire
Uil-OTS, Utrecht University
Nicole.Gregoire@let.uu.nl

Odijk (2005, 2003) developed a standard for the lexical representation of MWEs, which is called the Equivalence Class Method (ECM). The ECM was described for one type of MWE, viz. flexible idiomatic expressions, such as *het ijs breken* 'to break the ice'. In section 1 of this report, I discuss the ECM as proposed. Section 2 elaborates on one enhancement of the ECM, viz. the parameterized equivalence classes (PECs).

What is not addressed in this report, but will be discussed in the next one, is the specification of dependency structures in the lexical entry of an MWE, the use of parameters in the conversion procedure, and the illustration of how the adapted standard representation can be incorporated into a system-specific representation.

1 The MWE Equivalence Class Method

In section 1.1 I give an overview of the ECM as proposed by Odijk. Section 1.2 describes a procedure to convert the standard representation into a system-specific representation, in this case a representation required by the Rosetta MT system (Rosetta, 1994).

1.1 The proposed standard

Instead of describing the structure of an MWE, the proposed standard requires that it is specified which MWEs have the same structure.

Odijk proposes that an MWE description should consist of the following parts:

1. An MWE pattern name: an identifier that uniquely identifies the structure of the MWE.

2. A list of MWE components (Component List: CL).

3. An example sentence that contains the MWE.

The equivalence classes are defined with the help of the MWE patterns, i.e. MWEs with the same MWE pattern belong to the same equivalence class. The CL takes the form of a sequence of strings, each string representing the lexicon citation of an MWE component. The order of the sequence is free, but the standard requires that the same order is used for each MWE in the same equivalence class. As for the example sentence, the standard requires that the structure should be identical for each example sentence within the same equivalence class.

Besides the MWE description, there must be a list of MWE pattern descriptions. Each MWE pattern description consists of two parts: (1) an MWE pattern, and (2) comments, i.e. free text in which the uniqueness of the MWE pattern is described.

In Table 1 and 2 an illustration is given of the proposed standard. Table 1 shows one MWE pattern description and Table 2 shows three instances of the same MWE equivalence class, i.e. with the same MWE pattern.

Table 1: List of MWE pattern descriptions.

MWE pattern	**comments**
MWEp1	verb taking a subject and a directional PP headed by a postposition and with an NP complement consisting of a determiner and a singular noun
MWEp2	...

Table 2: Three instances of the same MWE equivalence class.

MWE pattern	**ICL**	**Example**
MWEp1	de pijp uit gaan	Hij is de pijp uitgegaan
MWEp1	het schip in gaan	Hij is het schip ingegaan
MWEp1	de boot in gaan	Hij is de boot ingegaan

1.2 The conversion procedure

Odijk (2003) shows that given a class of MWE descriptions, representations for a specific theory and implementation can be derived. This is illustrated by applying the procedure to the examples given in Table 2, and deriving the representation required by the Rosetta MT system (Rosetta, 1994). The actual representation of the idiom *de pijp uitgaan* (lit. 'go out of the pipe', id. 'to die') in the Rosetta lexicon is given in Table 3.

Table 3: Representation of *de pijp uitgaan* in the Rosetta Dutch lexicon.

	Lexical item element	**Description**
	ga	stem
	$s_aV_00_ga	skey
1	⟨$s_prep1286700 $s_aN_00_pijp⟩	skeys of idiom parts
2	[vpid30]	idiom pattern
3	$s_id_depijpuitgaan	idiom skey (lit. 'go out of the pipe')
4	$m_id_depijpuitgaan	idiom mkey
5	"dood gaan"	idiom meaning description ('to die')

The table shows the stem of the Dutch verb *gaan* ('to go'), which is *ga* followed by the syntactic key or skey ⟨$s_aV_00_ga⟩. The skey is a unique identifier for the syntactic item. Next, the following properties for the idiom are listed: (1) a sequence of skeys for the non-head components of the idiom, (2) the idiom pattern, (3) an skey for the idiom as a unit, (4) a unique identifier (mkey) for each meaning of the idiom, and (5) a meaning description for each meaning. It must be noted that there is no skey for the article *de*, since in the Rosetta system the article is introduced syncategorematically.

The procedure to convert the proposed standard into the representation required by the Rosetta system consists of two parts: a manual part and an automatic part.

1.2.1 The manual part

The manual part has to be carried out once for each MWE pattern P and consists of five steps:

1. Select an example sentence for MWE pattern P, and have it parsed by the system, yielding the *Reference Parse* (RP).

2. Define a transformation to turn the RP into the MWE structure.

3. Determine the list of unique identifiers of the lexical items used in the MWE, using the derived MWE structure, yielding the *MWE Component ID List* (ICIL).

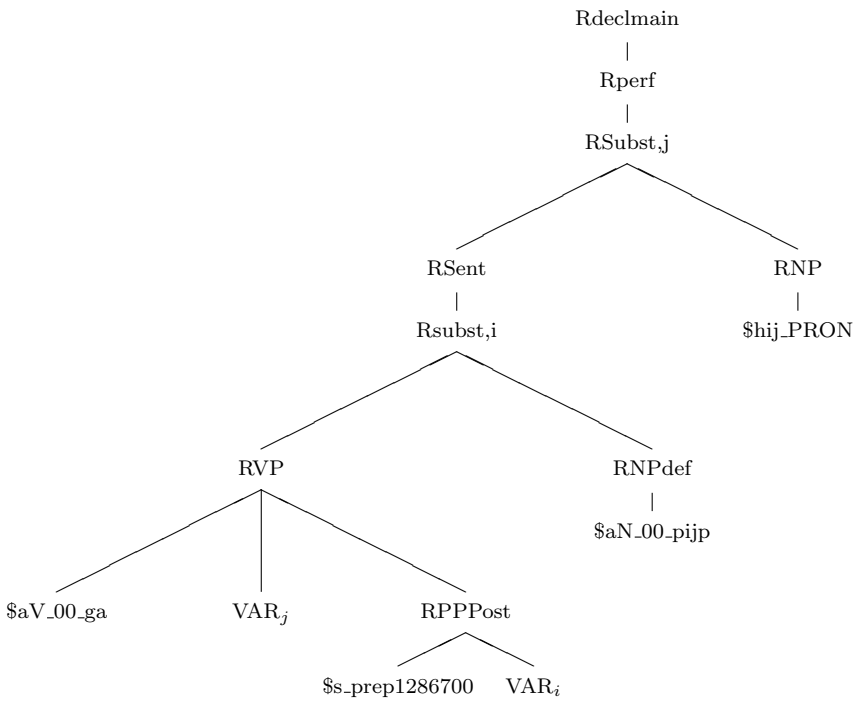
4. Define a transformation to relate ICL and ICIL.

5. Apply this transformation to the ICL, yielding the *Transformed ICL* (TICL) and check that the citation form of each lexical item equals the corresponding element on the ICIL.

The following illustration of the procedure is taken from Odijk (2004b, 2003). We apply the manual part to the MWE pattern *MWEp1* of the previous section.

1. We select the example sentence *Hij is de pijp uitgegaan* (lit. 'He has gone out of the pipe', id. 'He has died'). Parsing it by the Rosetta system yields the following syntactic D-tree (Reference Parse):

   (1)



2. The transformation to turn the RP into the structure of the MWE can be defined as follows: delete everything above the node containing the rule Rsubst,i.

3. We can use the skeys of the resulting tree to determine the ICIL: ⟨$V_00_ga,$sprep_1286700,$aN_00_pijp⟩, in this order.

4. The citation forms listed in the ICL (*de pijp uit gaan*) can be brought in correspondence with the ICIL by applying the transformation 1 2 3 4 → 4 3 2, i.e. delete the first element and reverse the remaining list.

5. Applying this transformation turns the ICL *de pijp uit gaan* into the TICL *gaan uit pijp*. The citation forms of the ICIL correspond to the elements on the TICL:

   - citation form ($aV_00_ga) = *gaan*
   - citation form ($s_prep1286700) = *uit*
   - citation form ($aN_00_pijp) = *pijp*

In this way we have obtained a procedure to convert MWEs of MWE pattern *MWEp1* represented in the standard format into the structure required in the Rosetta system.

1.2.2 The automatic part

After the manual work is done for one instance of an MWE class, the transformation of all other members of the equivalence class can be done in a fully automatic manner. The automatic part of the conversion procedure consists also of five steps:

1. Parse the example sentence of the MWE and check that it is identical to the RP for the example sentence used in the manual step, except for the lexical items.

2. Use the transformation defined in the manual step to turn the RP into the structure of the MWE.

3. Select the component IDs from the parse tree, in order to obtain the ICIL.

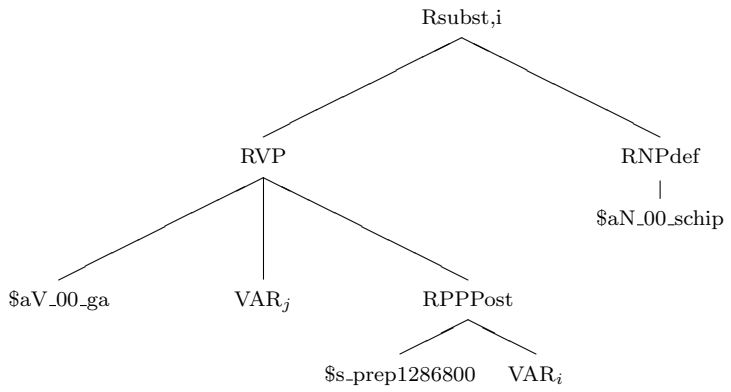
4. Apply the MWE component transformation to the ICL, in order to obtain the TICL.

5. Check that the citation form of each item in the ICIL equals the corresponding element on the TICL.

The automatic part is applied to each instance of the equivalence class. As illustration, we apply it to the MWE *het schip ingaan* (lit. 'to go into the ship', id. 'to have bad luck'). We follow the steps described above:

1. Parsing the example sentence *hij is het schip ingegaan* indeed leads to a syntactic D-tree that is identical to the one in (1), except for the skeys.

2. The transformation turns it into the following D-tree:

(2) 

]

3. ICIL is ⟨$aV_00_ga,$sprep_1286800,$aN_00_schip⟩, in this order.

4. The MWE component list transformation applied to the ICL *het schip in gaan* yields *gaan in schip*.

5. The citation form of each item on the ICIL equals the corresponding element on the TICL:

   - citation form ($aV_00_ga) = *gaan*
   - citation form ($sprep_1286800) = *in*
   - citation form ($aN_00_schip) = *schip*

1.3 Discussion

Several refinements of the proposed method are possible. In the remainder of this section I introduce one.

A potential problem of the ECM as proposed is the risk that the number of equivalence classes will run into thousands. In order to reduce the number of equivalence classes and to increase the number of members within each equivalence class, Odijk (2005, 2004a) introduced parameterized equivalence classes (PECs). The central idea behind this concept is that many MWE patterns describe structures that are for a large part identical.

In (3) we find the description of MWE pattern *MWEp1* for idioms such as *de plaat poetsen* (lit. 'to polish the plate', id. 'to bold').

(3)

MWEp1	expression headed by a verb taking a direct object NP that consists of a determiner and a singular noun, and taking one argument realized as the subject.

In the ECM another MWE pattern such as *MWEp2* in (4) is required for entries such as *de benen nemen* (lit. 'to take (away) the legs', id. 'to escape').

(4)

MWEp2	expression headed by a verb taking a direct object NP that consists of a determiner and a plural noun, and taking one argument realized as the subject.

And also another MWE pattern is required for MWEs such as *het varkentje wassen* (lit. 'to wash the little pig', id. 'take care of that'):

(5)

MWEp3	expression headed by a verb taking a direct object NP that consists of a determiner and a diminutive singular noun, and taking one argument realized as the subject.

The only difference between the three MWE patterns is the form of the noun it requires. The use of PECs reduces the number of MWE patterns, i.e. instead of four different unrelated MWE patterns *MWEp1...MWEp4*, one might assume a single MWE pattern *MWEp5* that takes two arguments (parameters), one to specify the number of the noun, and one to specify whether the diminutive form should be used.

Reducing the number of MWE patterns means reducing the number of equivalence classes. As a result, the number of MWEs that have to be dealt with manually minimizes, whereas the number of MWEs that can be incorporated into an NLP system in a fully automatic manner increases.

In the next section I discuss the use of the parameterized equivalence classes in detail.

2 Parameterized Equivalence Class Method

In this section I elaborate on the parameterized equivalence classes. In section 2.1 I present an overview of the parameters used in this research. The representation of parameters is discussed in section 2.2, and an overview of how parameters must be interpreted is given in section 2.3.

2.1 An overview of parameters

In this subsection I discuss the various aspects of the individual components within an MWE that we want to parameterize. These aspects include the form of a component and the way a component must be realized within a constituent. Recall that the main goal of parameterizing the equivalence classes is to reduce the number of classes, yielding less manual work in the conversion procedure.

In this approach parameterization is described with two terms, viz. *parameter category* and *parameter*:

Parameter Category The *parameter category* refers to the aspect we want to parameterize. Table 4 shows the parameter categories distinguished in this research.

Table 4: Overview of parameter categories.

category	**PC**	**description**
determiner	*dfrm*	the form of the determiner
noun	*nnum*	the number of the noun
	nfrm	the form of the noun
adjective	*afrm*	the form of the adjective
preposition	*ppos*	the way the preposition must be realized
verbs	*vfrm*	the form of the verb

Parameter The *parameter* is the value a parameter category takes. In Table 5 an overview is given of the different parameters divided by parameter category.

Table 5: Overview of parameters.

PC	parameter	description	PC	parameter	description
dfrm	DEF	definite	*nnum*	SG	singular
	INDEF	indefinite		PL	plural
	EMP	no determiner	*nfrm*	POS	positive
	DEMP	demonstrative pronoun		DIM	diminutive
	PNP	possessive NP		EINF	-e inflection
	SBP	subject bound possessive pronoun			
	OBP	object bound possessive pronoun			
vfrm	INF	infinitive	*afrm*	NORM	normal
	PART	particle verb		COMP	comparative
	PRESP	present participle		SUP	superlative
	PASSP	passive participle			
ppos	PREP	preposition			
	POST	postposition			

In addition to the parameters in Table 5, I introduce an optional parameters SPEC(ial).[1] The interpretation of this parameter is described in the Parameter Interpretation Manual in Appendix B.

2.2 Representation

Parameters are represented in the C(omponent) L(ist) of an MWE entry. Each MWE component is represented in the *canonical form*. In this research, the canonical form of the determiner is either the definite article *de* or *het*, depending on the gender of the noun.[2] VAR is used in cases where the complement of a pre-/postposition is a variable.

The parameters are realized between square brackets directly on the right of the item they parameterize. Example (6) shows a CL with parameter categories of an MWE that consists of a preposition-determiner-adjective-noun-verb.[3] The parameter category attached

[1] Label may be a better name for SPEC than parameter, since there is no opposite value and no corresponding parameter category. To avoid confusion about the term, I call SPEC a parameter.

[2] The gender of the noun is not specified.

[3] A p_slot[ppos] always precedes its complement in a CL, even if the *p_slot* must be realized as a

to the *v_slot* is optional, i.e. this parameter category is only represented if the *v_key*[4] is not the head of the expression.

(6) p_slot[ppos] d_slot[dfrm] a_slot[afrm] n_slot[nnum][nfrm] v_slot*[vfrm]*

Within the CL of an MWE entry, the slots are replaced with the canonical form of the corresponding lexical item and parameters are substituted for the parameter categories.

Some MWEs have multiple variants, e.g. *het varkentje wassen* and *dat varkentje wassen* (lit. 'to wash the/that little pig', id. 'to take care of the/that problem'). The determiner in this MWE can both take the form of a definite article and the form of a demonstrative pronoun. To account for such cases, we allow more than one parameter for each parameter category. Multiple parameters attached to one lexical item are divided with a slash (/).

In Table 6 we find some examples of CLs with parameters. It must be noted that these examples do not necessarily occur in the same equivalence class. In the next section I discuss the interpretation of the parameters.

2.3 Interpretation

In order for a linguist to understand what to do with the different parameters, I introduce the Component List Manual (CLM) and the Parameter Interpretation Manual (PIM). The CLM is a manual that basically shows what is in Table 4, but in more detail. The CLM is given in Appendix A.

The PIM is a manual in which it is specified how each parameter should be interpreted, i.e. how the individual components must be realized in order to get the MWE. Appendix B describes the PIM.

---

postposition.

[4] *x_key* is used as an abstract term for the canonical form of an MWE component.

Table 6: The CLs of some MWEs extended with parameters.

expression	**CL**
de plaat poetsen	de[DEF] plaat[SG][POS] poetsen
de benen nemen	het[DEF] been[PL][POS] nemen
een bok schieten	de[INDEF] bok[SG][POS] schieten
het varkentje wassen *dat varkentje wassen*	het[DEF/DEMP] varken[SG][DIM] wassen
wortel schieten	de[EMP] wortel[SG][POS] schieten
de pijp uitgaan	uit[POST] de[DEF] pijp[SG][POS] gaan
op de fles gaan	op[PREP] de[DEF] fles[SG][POS] gaan
zijn ogen uitkijken	het[SBP] oog[PL][POS] {uit}kijken[PART]
iemand op zijn vingers kijken *iemand op de vingers kijken*	op[PREP] de[OBP/DEF] vinger[PL][POS] kijken
iemands hart breken	het[PNP] hart[SG][POS] breken
het hart van iemand breken	het[DEF] hart[SG][POS] van[PREP] VAR breken
zijn beste beentje voorzetten	het[SBP] goed[SUP] been[SG][DIM] voor zetten
de helpende hand bieden	de[DEF] helpen[PPRES] hand[SG][POS] bieden
zich in de vingers snijden	zich in[PREP] de[DEF] vinger[PL][POS] snijden
buiten zichzelf zijn	buiten[PREP] zichzelf zijn
iets laten zien	laten zien[INF]
met iets opgescheept zitten	met[PREP] VAR opschepen[PASSP] zitten
iets ten tonele voeren	ten toneel[SG][EINF] voeren

A Component List Manual

This Component List Manual (CLM) describes how the Component List (CL) of an MWE entry must be read. The CLM should be used in combination with the Parameter Interpretation Manual given in Appendix B.

Each MWE component is represented in the *canonical form*. In this research, the canonical form of the determiner is either the definite article *de* or *het*, depending on the gender of the noun.[5] VAR is used in cases where the complement of a pre-/postposition is a variable.

Parameters are attached to the right of the component and are surrounded by square brackets. The parameter attached to the *v_key* is optional, i.e. this parameter is only realized if the *v_key* is not the head of the expression. The order within the CL depends on the equivalence class to which the MWE entry belongs.

Occurrences of representations within a CL are listed below in bold and/or small capitals:

- *x_key* refers to the canonical form of an MWE component with the syntactic category *X*.
- The parameter category can be found between the square brackets, on the right of the *x_key*.

Furthermore a description is given of how the representation must be read or interpreted.

d_key[*dfrm*]
The parameter attached to *de* or *het* refers to the form of the determiner.

a_key[*afrm*]
The parameter attached to a component with the category A refers to the form of this component.

[5]The gender of the noun is not specified.

n_key[*nnum*][*nfrm*]

The parameters attached to a component with the category N refer to (in this order):

1. the number of this component
2. the form of this component

v_key[*vfrm*]

The optional parameter attached to a component with the category V refers to the form of this component.

p_key[*ppos*]

The parameter attached to a component with the category P indicates whether this component must be realized to the left or to the right of its complement. It must be noted that this only accounts for Ps that take a complement.

p_key

The *p_key* without a parameter must be realized as an intransitive adposition, i.e. a P that does not take a complement.

zich

The canonical form *zich* refers to a weak reflexive, that must be bound to the subject. According to the person and the number of the subject, *zich* must be realized as *me*, *je*, *zich*, or *ons*.

zichzelf

The canonical form *zichzelf* refers to a strong reflexive, that must be bound to the subject. According to the person and the number of the subject, *zichzelf* must be realized as *mezelf*, *jezelf*, *zichzelf*, or *onszelf*.

[P1/P2]

The slash (/) is used to specify multiple parameters of the same parameter category. Specifying multiple parameters means that the *x_key*, to which these parameters are attached,

can take different forms. As a result, multiple MWEs must be derived from the MWE entry.

{part}verb

Particles are the separable part of separable compound verbs, such as opbellen ('to call (up)'), aanstaren ('to stare at'), uitleggen ('to explain'). In Dutch, the particle and the verb (with which the particle forms the particle verb) are written as one word when they are adjacent, but can also occur separated with other constituents intervening. In the CL, particle verbs are written as one word. To indicate which part of the particle verb is the particle and which part is the verb, the particle is surrounded by curly brackets ({ }).

B Parameter Interpretation Manual

This Parameter Interpretation Manual (PIM) describes how parameters must be interpreted. The PIM is implementation neutral, i.e. it is a guideline for linguists to help them to incorporate parameters into a specific system without describing a specific system. The ability to handle the parameters used in the standard, varies from system to system. A specific system that lists lexical entries with their canonical form and that derives the full form using rules, will most likely profit more from the parameters, than a system that lists lexical items in their full form, since they probably do not have any form derivation rules ready.

The interpretation is formalized as much as possible, i.e. I tried to avoid detailed descriptions. The interpretation of a parameter consists of either (1) a list of words, or (2) a rule/short description, or (3) both.

Table 7 shows each parameter and its interpretation. It must be noted that this list also contains parameters that do not occur in the Component List (CL) of an MWE entry, but are only used to refer to. The notation of the interpretation is as follows:

- The words between the [ ] are fully lexicalized components that can be substituted for the corresponding representation in the CL.
- $\rangle$ means 'depending on'. For example, the parameter DEF indicates that the determiner must be realized as *de* or *het* depending on the number and the form of the noun it is attached to.
- =*parameter* refers to another parameter in the table.

Special attention is paid to the parameter SPEC.

The parameter SPEC

The parameter SPEC is an optional parameter and is used in special cases where the full form of a component in an MWE is the colloquial form of a word, i.e. the form that is generally not used in formal speech or writing. In these cases the full MWE form is listed in the CL followed by the parameter SPEC. Examples of colloquial word forms in English are *gonna* and *ain't*. Two examples of Dutch MWEs that contain a word with the colloquial form are given in (7). It must be noted that only the colloquial form can be used when the

expression is used idiomatically. Substituting the normal forms *rode* and *dode* for *rooie* and *dooie*, respectively, changes the meaning of the expression in (7).

(7) a. over de **rooie** gaan
over the red-one go
'flip one's lid'

b. op zijn **dooie** akkertje
on his dead little-land
'dawdling'

Note on the parameters DEF and DEMP

In Dutch, the definite article/demonstrative pronoun of a plural noun is always *de*/*die*, and the definite article/demonstrative pronoun of a singular diminutive noun is always *het*/*dat*. This means that not all the parameters attached to the *n_key* are needed to derive the form of the *d_key* with the parameter DEF or DEMP:

1. If the *nnum* is PL, the *d_key* must be realized as *de*/*die*.
2. If the *nnum* is SG and *nfrm* is DIM, the *d_key* must be realized as *het*/*dat*.
3. If the *nnum* is SG and *nfrm* is POS, the *d_key* must be realized according to the canonical form:
   (a) *de*/*die* if the canonical form is *de*.
   (b) *het*/*dat* if the canonical form is *het*.

Note on the parameter EINF

The parameter EINF must be interpreted as an -e inflection operation that mostly applies to nouns that are the complement of a conflated preposition.[6] Examples are given in (8a) and (8b).

(8) a. iets ten **tonele** voeren
something on-the scene stage
'stage something'

b. iemand iets ter **ore** komen
someone something at-the ear come
'something comes to someone's attention'

[6]Conflated prepositions are prepositions combined with a definite article, such as *ten* and *ter*.

Table 7: Parameter Interpretation Table.

parameter	**interpretation**
POSS	[mijn][jouw][zijn][haar][ons][jullie][uw]
DEF	[de][het] 〉 n_key[*nnum*][*nfrm*]
INDEF	[een]
EMP	[ ]
DEMP	[die][dat] 〉 n_key[*nnum*][*nfrm*]
PNP	=POSS the full form of *d_key* is a possessive N
SBP	=POSS 〉 number and gender of the subject
OBP	=POSS 〉 number and gender of the object
SG	the full form of *n_key* is *n_key*
PL	the full form of *n_key* is the plural form of *n_key*
POS	the full form of *n_key* is *n_key*
DIM	the full form of *n_key* is the diminutive form of*n_key*
EINF	-e inflection of *n_key*
NORM	inflection of *a_key* 〉 d_key[*dfrm*]
COMP	the full form of *a_key* is the comparative form of *a_key*
SUP	the full form of *a_key* is the superlative form of *a_key*
PREP	The *p_key* is realized as a preposition, i.e. the complement of the *p_key* must be realized on the right of the *p_key*
POST	The *p_key* is realized as a postposition, i.e. the complement of the *p_key* must be realized on the left of the *p_key*
PART	*v_key* is a particle verb that is represented as {*particle*}*verb*
INF	the full form of *v_key* is the infinitive form of *v_key*
PRESP	the full form of *v_key* is the present participle form of *v_key*
PASSP	the full form of *v_key* is the passive participle form of *v_key*

References

Odijk, J. (2003), Towards a standard for multi-word expressions. ISLE Project Report, http://lingue.ilc.cnr.it/EAGLES96/isle/clwg_doc/ISLE_D6.1.zip.

Odijk, J. (2004a), A proposed standard for the lexical representation of idioms, *EURALEX 2004 Proceedings*, Université de Bretagne Sud, pp. 153–164.

Odijk, J. (2004b), Multiword expressions in NLP, Course presentation, LOT Summerschool, Utrecht.

Odijk, J. (2005), Standard lexical representation of multi-word expressions, http://www-uilots.let.uu.nl/irme.

Rosetta, M. (1994), *Compositional Translation*, Kluwer Academic Publishers, Dordrecht.