TIMEML: An ontological mapping onto UIMA Type Systems

Del Gratta Riccardo, Caselli Tommaso, Nilda Ruimy and Nicoletta Calzolari
Istituto di Linguistica Computazionale
Consiglio Nazionale delle Ricerche
via Moruzzi 1 -56124- Pisa, Italy
{riccardo.delgratta,tommaso.caselli,nilda.ruimy,nicoletta.calzolari}@ilc.cnr.it

Abstract

We present *TeR*, an UIMA Type System (Ferrucci and Lally, 2004) for event recognition, for temporal annotation in an Italian corpus[1]

We map each TIMEML category (Pustejovsky et al., 2006) to one or more semantic types as they have been defined in the SIMPLE-CLIPS ontology (Ruimy et al., 2003). This mapping presents some advantages, such as the orthogonal inheritance that an event can acquire when derived from the ontology and a clearer definition of semantic roles when carried out by events.

The mapping is implemented by means of a FINITE STATE AUTOMATON which uses semantic information collected from the SIMPLE-CLIPS ontology to analyze natural language texts.

1 Introduction

Temporal information has become one of the key points in Computational Linguistics and Semantics Research fields. A lot of studies pointed out that the understanding and treatment of "time" in texts play a crucial role both theoretically, (e.g: for "bridging anaphora" resolutions and phrase structures), and from an applicative perspective, (e.g: Open Domain Question-Answering, Information Retrieval, Information Extraction, Semantic Web, etc.) (Saurí et al., 2005).

In natural language texts, events are strongly anchored in time. It is by using time and temporal relations between events that, as human, we can reason on changes of certain situations (Hobbs and Pustejovsky, 2003). Temporal relations among different entities represent the core elements to describe the temporal ordening of a situation. Moreover, temporal ordening understanding is also responsible for reasoning.

By annotating events we can draw a kind of map over the text itself which makes easier the access to the information through temporal context rather than keywords.

In this article, we present an "UIMA-based" resource interoperability achieved by integrating the PAROLE-SIMPLE-CLIPS lexical resource (Ruimy, 2006) with both TIMEML annotation guidelines and rule-based heuristics. One of the main advantages of using the UIMA platform is that it allows the "embedding" of already existing resources in the framework, rather than the implementation of new ones and the possibility to easily integrate additional tools in an "IDE" such as *Eclipse*.

[1] More information about corpus used, and its characteristics, see (Caselli et al., 2007)

2 Linguistic issues

TIMEML is one of the most complete annotation scheme for temporal annotation and it has been recently proposed as an ISO standard. Its specification languages allows the identification of events and states (e.g: go, try, peace, on board ...), temporal expressions (e.g: December 25th, four years ...), and temporal links among these entities (e.g: after, during ...).

For the purpose of this work, we concentrate only on the event category. TIMEML employs a rather pre-theoretical notion for defining events and states, i.e. as something which happens or occurs, and as situations which hold or obtain to be true. Events are not classified using the particular meaning of the verb which describes them, but through the use of semantic information encoded in the events themselves. Once identified, events are classified in one of the following categories:

- REPORTING
- PERCEPTION
- ASPECTUAL
- I_ACTION
- I_STATE
- STATE
- OCCURRENCE

This classification is useful since it is language independent and relevant for "characterizing the nature of an event as being irrealis, factual, possible, reported, intensional" (Saurí et al., 2005).

Our approach to event recognition and classification, is to link -or better to map- each of the above seven categories to one or more semantic types as they have been defined in the SIMPLE-CLIPS ontology. This mapping provides semantic information because each event is associated with a lexical entry from which it inherits other semantic information, according to the orthogonal inheritance principle (Pustejovsky and Boguraev, 1993).

3 Automatic TIMEML tagging

This section analyzes two of the various attempts performed to semi-automatically recognize TIMEML category (section 3.1) and describes our proposal (section 3.2).

3.1 Background

Attempts to recognize TIMEML categories have been performed by using ontology semantic types (Caselli et al., 2007), or by using TimeBank and machine learning approaches (Boguraev and Ando, 2006).

The strategy followed by Caselli et al. (2007) is to (manually) check whether a word sense[2] denotes an event and which is its ontological semantic type. Heuristics have been used to check the current word sense against its semantic and syntactic properties as defined in the lexical resource for its classification.

The strategy followed by Boguraev and Ando (2006) is a hybrid approach using both a finite state grammar for temporal expressions and a machine learning technique trained on the TimeBank and unannotated corpora. The finite state grammar is embedded in a shallow parser, while the machine learning algorithms implement novelty in learning from unannotated corpora.

3.2 Our proposal

Event recognition has recently reached quite good levels although improvements can still be obtained.

This section explains our approach to (semi-automatically) implement a **T**imeML **E**vent **R**ecognizer, henceforth *TeR*.

Our idea is formally based on the work of Caselli et al. (2007) but the strategy we follow is the opposite.

We use the SIMPLE-CLIPS ontology to answer the question:

when does a word sense denote an event?

In the SIMPLE-CLIPS ontology, each word sense is classified in terms of the semantic type it belongs to. Word senses which belong to the SIMPLE-CLIPS event type system are collected into lists according to the *type* of the event. These lists are then processed to mark-up word senses with the correct TIMEML category which is "suggested", on the one hand, by the semantic type to which the senses belong to, and on the other, by rule-based heuristics. In addition, the use of the PAROLE-SIMPLE-CLIPS

[2] By word sense we mean the sense of the word currently analyzed. Word sense disambiguation is a crucial aspect in (Caselli et al., 2007) work. Also in our proposal this issue has to be addressed.

lexical resource allows TIMEML categories to be enriched with a lot of morphosyntactic and semantic features, directly inherited from the resource.

The chance to uniquely classify events following TIMEML specifications represents an important step toward the implementation of algorithms capable of managing a strong automatic treatment of texts.

The UIMA platform is a focus in our event recognition approach, since it is responsible for providing both the final mapping between the SIMPLE-CLIPS semantic types and the TIMEML categories via rule-based algorithms and heuristics implementation and the common platform on which those resources are integrated.

Our goal is to define a set of UIMA Type Systems useful to identify TIMEML categories in texts.

4 TeR, a TIMEML Event Type System

TeR is a **T**imeML **E**vent **R**ecognizer implemented as an UIMA TYPE SYSTEM used to tag word senses which denote events. The TeR is built upon the PAROLE-SIMPLE-CLIPS lexical resource from which it inherits syntactic and semantic information. This additional information is attached to the word sense and handled as UIMA Type System features of the TeR, which is promoted to be a "complex collector" of several linguistic information as well as the "classifier" of the TIMEML category.

As explained in section 6, TeRs are built by integrating different resources, which are described in the following subsections. Section 4.1 is a brief introduction to the SIMPLE-CLIPS ontology; section 4.2 presents the UIMA Type System hierarchy and features; section 4.3 is a description of the TeRs as "complex collectors" of information and, hence, complete annotators.

4.1 The SIMPLE-CLIPS Ontology

In the PAROLE-SIMPLE-CLIPS lexical database, at the semantic layer of information, lexical units are structured in terms of a semantic type system and are characterized and interconnected by means of a rich set of semantic features and relations. The type system structure consists of 157 language-and domain-independent semantic types designed for the multilingual lexical encoding of concrete and abstract entities, events and properties.

The SIMPLE-CLIPS ontology, as already stated, implements the principle of orthogonal inheritance, whereby multidimensionality is captured by qualia roles[3] (Pustejovsky, 1995) which define the distinctive properties of semantic types and differentiate their internal semantic constituency.

According to the philosophy governing the SIMPLE-CLIPS ontology, a semantic type is the repository of a structured set of semantic information about a lexical unit. In the lexical database, predicative word senses are assigned a predicate-argument structure. Predicative information consists in the description of the argument structure in terms of predicate arity, semantic role and semantic constraints of each argument[4]. It is worth noting that the encoding of restrictions on arguments entails that the lexical resource provides information not only on word senses but also on their semantic context, which is a useful information for event classification according to the TIMEML specifications.

4.1.1 Event tree structure

In SIMPLE-CLIPS ontology, 59 different event types have been defined. These event semantic types are organized in the typical tree structure: we can identify 7 main (root) event types, each one subsuming a certain number of sub-events.
This event tree structure is an "*is-a*" relation with edges and nodes arranged in a tree structure. In the perspective of the FINITE STATE AUTOMATON outlining (see section 5.6), the distance of the node from the upper root[5], as well as its direct ancestor are relevant for rule-based heuristics implementation.

4.2 Defining UIMA Type Systems for TeRs

As a framework for our common platform, we adopted the UIMA Architecture. UIMA provides both the integration of single NLP components, PRIMITIVE ANALYSIS ENGINES, and NLP pipelines deployment facilities, AGGREGATE

[3] In the framework of the SIMPLE-CLIPS ontology model definition, each of the four *qualia* roles has been promoted as top node of a hierarchy of semantic relations which altogether form the *Extended Qualia Structure*.

[4] Constraints are expressed in terms of semantic type, features, or "notions" combining these different expressive means.

[5] Here, the upper root node is the (generic) "Event" .

ANALYSIS ENGINES. The former can be either existing resources or resources built from scratch within the UIMA framework, while the latter represent a NLP workflow defined on the integration of single components via descriptor fence.

The ANNOTATION TYPE, i.e. the TeR upon wich a PRIMITIVE ANALYSIS ENGINE is implemented, that we propose consists of three different kinds of features[6], as reported in table 1:

UIMA TeR feature	**Type of information**
TIMEML Category	Temporal information
Part of Speech Grammatical information Lemma	Morpho-syntactic information
Semantic Type Features Relations	Semantic information

Table 1: TeRs features

Our idea is to set the TeR feature values with information stored in PAROLE-SIMPLE-CLIPS lexical database. Since PAROLE-SIMPLE-CLIPS is a rich resource, each TeR can be enriched with morphological, syntactic and semantic information from the resource, thus becoming a multidimensional object.

We define a super type, *SimpleTeR*, which directly inherits from the *uima.tcas.annotation* type and implements all the features and relations shared among different event types[7].

We define seven different TeRs, one for each TIMEML category. These TeRs inherit from the *SimpleTeR* and record semantic and morphosyntactic information of terms they are annotating as UIMA Type System feature values. The TeRs behave both as standard annotators (including semantic aspects) and as temporal annotators.
Figure 1 shows the hierarchy defined for TeRs. The sentence annotator and its relevance is explained in section 5.7.

[6]For the sake of clarity, we remind that a feature, in UIMA language, is an attribute name-value pair.

[7]At the top level only the formal relation and the link to the Entity class are implemented.

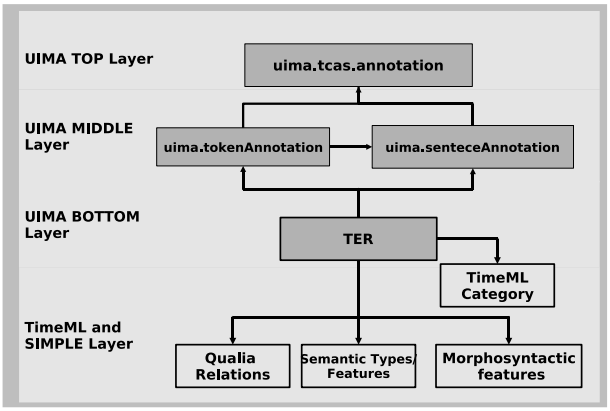

Figure 1: TeRs Hierarchy

4.3 TeR is a "complex collector" of information

An important side effect of retrieving lists of word senses from a lexical resource is that these word senses inherit both semantic properties defined for the semantic type they belong to and morphosyntactic features from the syntactic unit. TeRs have been defined to manage all this complex bundle of information using specific features.
Annotating texts with TeR allows users to perform a multilevel annotation by filling the (UIMA) feature values with semantic and morphosyntactic information which, in turn, enrich the TIMEML information itself.

5 Building UIMA Type Systems: From lexicon to annotators

This section explains some preliminary steps to be performed during the tuning of the lexical resource. Sections 5.1 to 5.3 address the cleaning of the lexical resource, while sections from 5.4 to 5.6 describe the classification of TIMEML categories and the algorithms used to map PAROLE-SIMPLE-CLIPS lexical units onto these categories.

5.1 Cleaning the PAROLE-SIMPLE-CLIPS database: preprocessing

The analysis of the resource shows that a word may have multiple senses and that one sense may have different syntactic behaviors and/or different subcategorization frames. All these ambiguities lead to the word sense disambiguation issue from the semantic perspective and/or to the co-textual analysis from the syntactic perspective.

The PAROLE-SIMPLE-CLIPS database preprocessing step consists in filtering out all event denoting words[8] which are linked to more than one sense and/or have more than one syntactic behaviour. This approach defines an "*a priori*" disambiguation, so that, at the beginning of the annotation process, only unambiguous words are processed.

5.2 Cleaning the PAROLE-SIMPLE-CLIPS database: evaluation of disambiguation

The above defined ambiguities weigh on the whole set of event denoting words in the percentage reported in table 2:

Type of Ambiguity	Percentage
Semantic	37%
Syntactic	21%

Table 2: Ambiguity Distribution

The percentage of syntactic ambiguity reduces to 17% for semantically unambiguous terms.

After preprocessing step, the annotation system is able to *automatically* recognize up to 53% of unambiguous events in texts. By unambiguous event, we mean an event denoted by a word which has one and only one sense and one and only one syntactic behavior.

5.3 Cleaning the PAROLE-SIMPLE-CLIPS database: list of words output format

For unambiguous words we have decided the following output structure:

Morpho-Syntactic information	Semantic information

Table 3: output file structure

In table 3, morphosyntactic information consists of morphological units, subcategorization frame, lemma, inflected forms and grammatical features; semantic information is the set of semantic features and relations which characterize a semantic type.

[8] An event denoting word is a word belonging to the SIMPLE-CLIPS event type system.

This output format is relevant to assign the correct TIMEML category both when only semantic information is needed and when also syntactic criteria are used to select the TIMEML category (see section 5.4). It is worth noting that this format is coherent with the TeR Type System defined in section 4.2.

5.4 Classification of TIMEML categories

Mapping the SIMPLE-CLIPS ontology over the TIMEML categories defines a correlation between the former and the latter. The relation between the ontological types and the TIMEML classes is "one to many". This is due to the fact that the TIMEML categories depend on several criteria consisting either of semantic clues or of a mixture of semantic and syntactic ones.

Heuristics necessary to uniquely identify the final TIMEML category have been implemented to improve the mapping.

All experiments performed suggest that a clear approach must be followed to map the SIMPLE-CLIPS ontology to the TIMEML categories: first semantic information is analyzed and then co-textual information (i.e. verb forms, argument structure or syntactic dependencies) is applied to assign the right TIMEML class. We classify the seven TIMEML categories according to the following rules:

TIMEML Category	Information
REPORTING PERCEPTION ASPECTUAL	Semantic Criteria, including lexical meaning
I_ACTION I_STATE STATE	Semantic, including lexical aspects, plus syntactic criteria
OCCURRENCE	Default exit [9].

Table 4: TIMEML categories and rules

5.5 Coarse-grained mapping

The PAROLE-SIMPLE-CLIPS resource is implemented as a relational database. Each object, implementing semantic properties described in section 4 and other morphosyntactic information, is a specific

[9] In the following we will see that a FS grammar built over SIMPLE-CLIPS ontology always has an "OCCURRENCE" as default exit when no other TIMEML category could be assigned.

table. The structure of the tables and their values define both how to map the TIMEML category onto word senses and how TeRs are built accordingly.

This subsection is dedicated to the outline of the FINITE STATE AUTOMATON (FSA) responsible for the TeRs definition and implementation in a coarse-grained mapping scenario.

Following Caselli et al. (2007), we can implement basic rules which, essentially, rely on the event tree structure. As a starting point of the mapping process we consider only the 7 root-events. Fine-grained investigation over sub-events is performed at a deeper level of analysis, when heuristics are implemented (see section 5.6).

The first "coarse-grained mappings" implemented concern the *Phenomenon* and the *State* semantic types: all word senses belonging to these types and their subtypes are mapped to the TIMEML category of OCCURRENCE and STATE respectively (see table 5):

Event	TimeML	Rule
Phenomenon	OCCURRENCE	Stop at root level
State	STATE	Stop at root level

Table 5: Coarse-Grained Mapping for Phenomenon and State semantic types

The FSA rule can be read as follows:

- `output_step_0=Extract words belonging to Phenomenon (State) from SIMPLE-CLIPS;`
- `output_step_1=Match strings in texts with the output_step_0;`
- `Exit_FSA=Tag output_step_1 with OCCURRENCE (STATE);`

Figure 2 shows the FSA for the coarse-grained mapping.

5.6 Fine-Grained Mapping

Fine-grained mapping combines both linguistic and heuristic-based techniques to better assign the TIMEML category to a given word sense.

In a fine-grained mapping scenario, FSA uses the rules directly resulting from the heuristics, in which

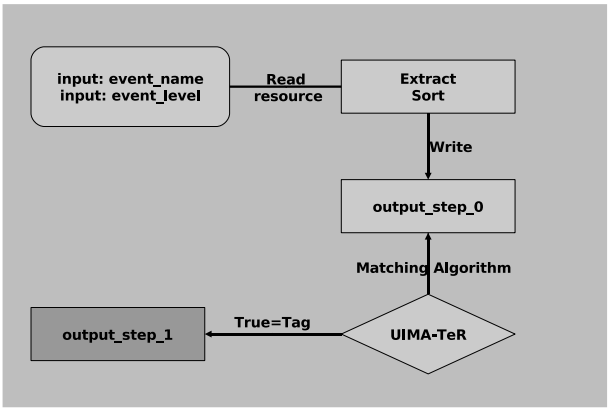

Figure 2: FSA for the coarse-grained mapping.

both semantic and co-textual criteria are used to build guidelines in a hierarchical order of application, whereby the semantic information is dealt with beforehand.
The strategy can be summarized as follows:

- `output_step_0=Create lists of words for each class of events and sub-events;`
- `wip`[10]`_step_0=Add semantic and mophosyntacic information each word in output_step_0;`
- `wip_step_1=Apply heuristics;`
- `output_step_1=Extract from output_step_0 only words uniquely mapped onto TIMEML categories;`

The fine-grained mapping is essentially based upon wip_step_1. In this step, we apply heuristics to each list of words. The internal rules of heuristics allow FSA to take decisions about the TIMEML category a given word sense should belong to. Co-textual analysis and predicative structures in prepositional phrases are key points in fine-grained mapping, since FSA is also driven by the complex phrase structure (see section 5.7).

5.7 Co-Textual analysis and sentence annotator

In order to be as compliant as possible with TIMEML specifications and to manage event variability, we have to consider, also, a portion of text

[10] wip means work in progress.

surrounding the event and the actual realization of the event itself.

Some heuristics check whether an event has another event as its proper argument. To be able to manage this co-textual analysis we implement the "window capability" of the UIMA framework, i.e. the number of tokens which are analyzed in the same annotation process. Window capability allows UIMA to span a large part ot text. The main token of the window is the event denoting word sense and other tokens in the same window are analyzed by an automaton responsible for semantic and morphosyntactic recognition. In this scenario single tokens are relevant not only by themselves, but also for their interactions with other tokens. Since the sentence is a coherent set of tokens, we chose one single sentence as the spanned text in the "window". Within a sentence, the FSA recognizes different syntactic behaviors and semantic restrictions on predicative structure and heuristics can be completely applied.

6 First prototype of UIMA Type System annotation tool

A first prototype for UIMA Type System has been built to manage TIMEML categories which do not need the window size in the text analysis, i.e. they do not need the co-textual analysis to be assigned to the word sense. For example, the *Phenomenon* semantic type belongs to this kind of categories. Input text is passed to UIMA and translated into an object, the CAS (Götz and Suhre, 2004), which contains both physical and metadata of the text itself. The CAS ***initialize*** and ***process*** methods are responsible for cleaning and setting up the PAROLE-SIMPLE-CLIPS resource and for the mapping between the unambiguous word senses and the TIMEML categories respectively.

A second step is the development of the co-textual automaton as independent software. This automaton is called by the UIMA framework every time a window capability is required. The sentence annotator represents the basic step to implement the co-textual automaton. Sentence annotator and TeR are "aggregated" so that, within a single sentence, every token is analyzed by a morphosyntactic parser and the information retrieved is sent back to the UIMA platform to be handled by the ***process*** method of the AGGREGATE ANALYSIS ENGINE responsible to apply heuristics. Figure 3 below shows how the interoperability between the PAROLE-SIMPLE-CLIPS resource and TIMEML specification is implemented. The layout of the figure is based on the Language Grid Service Ontology. The arrows point to objects which are the subjects of the properties: for example in "PAROLE-SIMPLE-CLIPS *usedBy* UIMA FW", the arrow points towards PAROLE-SIMPLE-CLIPS.

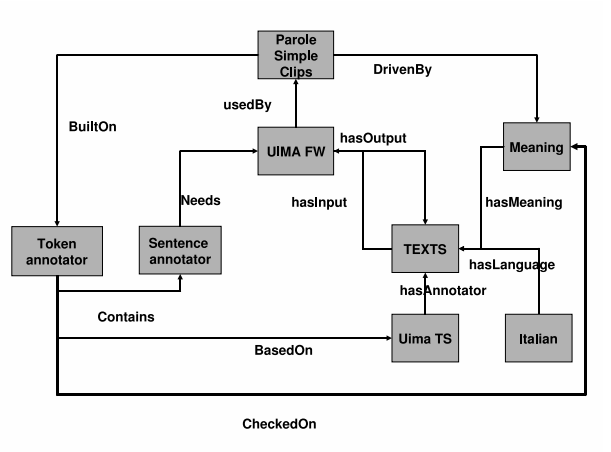

Figure 3: Work Flow

7 Testing the platform

This section briefly describes some test cases that have to be performed after the first prototypes have been deployed.

The corpus to be annotated is the one used by Caselli et al. (2007) for their manual mapping of events. Our results will be checked against their results in terms of numbers of events identified, percentage of agreement on events and overall agreement on classification (K-statistic).

8 Managing the disambiguation

As explained in section 5.2, a word may have different senses.

Since the TIMEML categories are mapped to words via their sense, the disambiguation problem is crucial in TIMEML mapping.

For example, the Italian word "distrutto" in the following sentences:

1. Edificio **distrutto**...[**Destroyed** building ...]

2. Le tue parole mi hanno **distrutto** [Your words **wrecked** me]

has two different senses: in the first sentence the sense is linked to the semantic type *Cause_change_of_state*, while in the second it is linked to the *Cause_experience_event*.

The word "distrutto" has to be mapped onto two different TIMEML categories:

- distrutto$(1) -> $ STATE
- distrutto$(2) -> $ OCCURRENCE

The FSA, in the fine-grained mapping scenario, is able to switch between these two possible senses according to their different syntactic behaviour.

9 Conclusion

We have presented TeR, a TIMEML Event Annotator modeled upon a rich lexical resource like PAROLE-SIMPLE-CLIPS. The "a priori" disambiguation formalized in the resource allows TeR to automatically tag up to 53% of words, since this is the percentage of unambiguous terms in the resource.

Resources interoperability is a focus in this project, and the UIMA Platform is the common framework used to integrate resources. This paper intends to contribute both to a UIMA TYPE SYSTEM standard and to a common framework for resource sharing and interoperability definition. Moreover, an operative workflow in the infrastructure is defined.

Strong links are established with the GrAF (Ide and Suderman, 2007) annotation framework, since the span feature of the GrAF is easily mapped on the begin-end features of the TeR, and with the NICT language grid project (Ishida, 2007), from which our prototype inherits the service ontology environment.

In addition, by adding these Type Systems to the UIMA platform, researchers can use them to add a new annotation layer to already existing corpora e. g. to TreeBanks.

References

Branimir Boguraev and Rie Kubota Ando. 2006. Analysis of timebank as a resource for time-ml parsing. In *Proceeding of LREC-06*, May 2006, Genova.

Tommaso Caselli, Irina Prodanof, Nilda Ruimy, and Nicoletta Calzolari. 2007. Mapping simple and timeml: improving event identification and classification using a semantic lexicon. In *GL2007: Fourth International Workshop on Generative Approaches to the Lexicon*, 10-11 May 2007, Paris.

David Ferrucci and Adam Lally. 2004. Uima: an architectural approach to unstructured information processing in the corporate research environment. *Nat. Lang. Eng.*, 10(3-4):327–348.

Thilo Götz and Oliver Suhre. 2004. Design and implementation of the uima common analysis system. *IBM Systems Journal*, 43(3):476–489.

Jerry Hobbs and James Pustejovsky. 2003. Annotating and reasoning about time and events. *Proceeding of the AAAI Spring Symposium on Logical Formalization of Commonsense Reasoning*, pages 74–82.

Nancy Ide and Keith Suderman. 2007. Graf: A graph-based format for linguistic annotations. In *Linguistic Annotation Workshop, ACL 2007*, Prague.

Toru Ishida. 2007. Nict, language grid & department of social informatics, kyoto university. *http://langrid.nict.go.jp*.

James Pustejovsky and Branimir Boguraev. 1993. Lexical knowledge representation and natural language processing. *Artif. Intell.*, 63(1-2):193–223.

James Pustejovsky, Jessica Littman, Bob Knippen, Robert Gaizauskas Andrea Setzer, and Roser Saurí. 2006. Timeml annotation guidelines. *http://www.timeml.org/site/publications/specs.html*.

James Pustejovsky. 1995. *The Generative Lexicon*. MIT Press, Cambridge.

Nilda Ruimy, Monica Monachini, Elisabetta Gola, Nicoletta Calzolari, Cristina Del Fiorentino, Marisa Ulivieri, and Sergio Rossi. 2003. A computational semantic lexicon of italian: Simple. *Computational Linguistics in Pisa, Istituto Editoriale e Poligrafico Internazionale*, pages 821–864.

Nilda Ruimy. 2006. A computational multi-layered italian lexicon for hlt applications. In *Proceedings XII EURALEX International Congress, Atti del Congresso Internazionale di Lessicografia*, volume 1, pages 221–227, Torino, 6-9 settembre.

Roser Saurí, Robert Knippen, Marc Verhagen, and James Pustejovsky. 2005. Evita: A robust event recognizer for qa systems. short paper. In *Proceedings of HLT-EMNLP 2005, 700-707*.