Semantic Annotation Using NKRL (Narrative Knowledge Representation Language)

Gian Piero Zarri[1]

Abstract. We suggest that it could be possible to come closer to the Semantic Web goals by using 'semantic annotations' that enhance the traditional ontology paradigm by supplementing the ontologies of concepts with 'ontologies of events'. We present then some of the properties of NKRL (Narrative Knowledge Representation Language), a conceptual modeling formalism that makes use of ontologies of events to annotate in great detail those 'narratives' that represent a very large percentage of the global Web information.

1 INTRODUCTION

As well known, the current state of Web technology — the 'first generation', or 'syntactic' Web — gives rise to serious problems when trying to accomplish in a non-trivial way essential tasks like indexing, searching, extracting, maintaining and generating information. These tasks would, in fact, require some sort of 'deep understanding' of the information dealt with: in a 'syntactic' Web context, on the contrary, computers are only used as tools for posting and rendering information by brute force. Faced with this situation, Tim Berners-Lee first proposed a sort of 'semantic Web' where the access to information is based mainly on the processing of the semantic properties of such information, and its extraction and rendering on the use of heuristics (inference rules) that make use of these properties. To realize the semantic Web vision, it becomes then necessary to find a way of describing such semantic properties in a computer-understandable way.

A natural way of 'saying something' about a multimedia document consists into 'annotating' it by including additional, descriptive information (metadata): annotation, in fact, is intended in general as the adding of meta-information to a document as to provide its generic 'enrichment'. A well-known example of this technique is given by the Annotea Project [1], where annotations are conceived as means of associating general remarks to an existing document, in the style of "The text in this document does not make much sense". Without questioning at all the concrete utility of such tools, we must remark that such a sort of generic annotation does not correspond exactly to the requirements for the semantic Web as proposed by Berners-Lee. Given, in fact, the 'deep understanding' requirements expounded before, 'annotation' must now be strongly understood as 'semantic annotation', intended therefore to convey, in some way, the actual 'meaning' of the document.

When we examine the 'standard' existing proposals in the 'semantic' annotation and metadata domains, it is evident that, very often, they can be hardly defined as 'semantic'. A well known project in this context is represented by the Dublin Core Initiative [2], based mainly on the use of a set of 15 metadata elements (title, subject, description, source, language, creator, publisher, date, type, format etc.). Starting from July 2000, these elements can be associated with 'qualifiers' [3] — like 'Is Part Of' and 'Has Part' (but, strangely, not 'Is Member of' or 'Has Member') — to allow additional levels of detail. Apart from the evident impossibility of describing the semantics of the Web making use of only 15 categories, we can note that a majority of these last deal mainly with the 'external identification framework' (title, creator, publisher…) and the 'physical structure' (format…) of the digital documents stored on the Web more than with a description of their true 'semantic meaning' [4].

Note that RDF (Resource Description Framework), see [5, 6] — a proposal for defining and processing WWW metadata that has developed by a specific W3C Working Group (WRC = World Wide Web Consortium) — is often associated to the Dublin Core thanks to the fact that the RDF description of this Core has been for long the only concrete, existing application of the RDF techniques. RDF is, however, only a tool that is independent from any particular metadata system it can implement — Dublin Core apart it is used, e.g., in two very different approaches to 'annotating' like Annotea and NKRL (see below). The RDF model, implemented in XML (eXtensible Markup Language), makes use of Directed Labeled Graphs (DLGs) where the nodes, that represent any possible Web resource (documents, parts of documents, collections of documents etc.) are described basically by using attributes that give the named properties of the resources: the values of the attributes may be text strings, numbers, or other resources. Initially, the model bore a striking resemblance to some early KR work on semantic networks; the (provisional) RDF specification, see [5], includes now more advanced KR constructs like the 'containers', i.e., tools for describing 'collections' of resources.

Metadata — at least in their Dublin Core connotation — means in practice keywords, which can be assimilated to low-level 'concepts' taken in isolation. A natural (and very popular today) extension of the metadata approach consists, therefore, in the use of concepts structured according to an 'ontology' to (try to) describe the 'semantics' of the Web.

Several knowledge representation languages based on the use of ontologies have been proposed and tested in a Semantic Web context, see, e.g., DAML+OIL [7, 8], SHOE [9], etc. All of these are able to offer inferential services based on the use of hierarchies of concepts, i.e., (basically) frame-like hierarchical structures where the nodes are represented by the formal definitions, thorough properties and axioms, of the important notions of the domain (concepts).

Making use of ontologies constitutes, undoubtedly, an important step towards true semantic-grounded utilization of the

[1] Centre National de la Recherche Scientifique (CNRS), 44 rue de l'Amiral Mouchez, 75014 Paris, France, email: zarri@ivry.cnrs.fr

Web; ontologies may not be sufficient, however, to fully render the semantic content of all of the Web resources. For example, a large part of the Web information that is of an industrial and economic interest consist often of 'narratives' about 'actions', 'facts', 'events', 'states' etc. that relate the real or intended behavior of some 'actors' (characters, personages, etc.), like news stories, corporate documents (memos, policy statements, reports and minutes), normative and legal texts, intelligence messages, medical records, etc. In this case, the simple description of concepts is not enough, and must be integrated by the description of the mutual relationships between concepts — or, in other terms, the description of the 'role' the different concepts and their instances have in the framework of the global actions, facts, events etc. Ontologies normally supply, on the contrary, only a quite static, rigid vision of the world, a taxonomy of pinned up, 'dead' concepts.

This paper would like then to suggest that it could be possible to come closer to the Semantic Web goals by making use of 'semantic annotations' that enhance the ontology paradigm by supplementing, in particular, the traditional ontologies of concepts with 'ontologies of events', i.e., new hierarchical structures where the nodes are now 'templates' that represent formally generic classes of elementary events like "move a physical object", "be present in a place", "produce a service", "send/receive a message", "introduce a change", etc. Templates represent then dynamic relationships between the basic concepts: they are fundamental for the correct rendering of narratives, especially when they are associated with the use of second order tools able to take into account the 'connectivity phenomena' (logico-semantic links, like CAUSE and GOAL) that, in a narrative situation, can exist between single narrative fragments.

In the following, we will present quickly, in Section 2. and the following, some of the main properties of NKRL ('Narrative Knowledge Representation Language'), see [10, 11], a language expressly designed for representing, in a standardized way, the 'meaning' of complex multimedia information like that typically found on the Web. NKRL has been used as 'the' modeling knowledge representation language for annotating narratives in European projects like Nomos (Esprit P5330), Cobalt (LRE P61011), Concerto (Esprit P29159), Parmenides (IST P39023), etc. It is used, e.g., in the current European Euforbia project (IAP P26505) to annotating and filtering 'questionable' Web sites according to a semantic-rich approach, see the next Section for some details. NKRL constitutes then for sure one of the most achieved and powerful solutions to the annotation problem.

Because of the space limitations, we will deal here only with the knowledge representation aspects of the formalism. For information on the natural language (NL) features — i.e., how to pass, automatically or semi-automatically, from the NL formulation of an 'event' to its corresponding NKRL representation — see, e.g., [12] and, for the work accomplished in the framework of the Concerto project, [13]. A recent paper on the NKRL high-level inference procedures is [14].

2 THE MAIN KNOWLEDGE REPRESENTATION TOOLS OF NKRL

In Euforbia, NKRL is used to associate with a Web site — when it is first inserted on the Web or at the moment of a major restructuring — an 'Euforbia label' that represents the semantic content of the *whole* site (i.e., the home page plus the associated pages). An Euforbia label includes three sections, where only the first is mandatory:

- the 'aim' section, i.e., a description of the main objectives of the site;
- the 'properties' section, i.e., a description of some characteristics of the site that could be interesting to register (like the fact that the site is a free or a paying one, the increment or decrement in the number of hints, the way of managing the site etc.);
- the 'sub-sites' section, i.e., a list of the associated sites with a short NKRL description of the main functions of each of them.

Table 1 reproduces an (extremely simplified) image of the aim section of the Euforbia label associated with the site "London Escort Agency, http://www.london-escort-agency.co.uk/"; the (intuitive) meaning of this symbolism is: "the London Escort Agency provides an escort service". We will then make use of the code of Table 1 to introduce the main properties of the NKRL language, see [10, 11] for additional details.

Table 1. A (very simple) example of NKRL code

```
c1)  PRODUCE
     SUBJ  (SPECIF london_escort_agency
            (SPECIF escort_agency uk_)):(london_uk)
     OBJ   escort_service_1
```

The NKRL knowledge representation tools are organized into four connected 'components', the definitional, enumerative, descriptive and factual component.

The 'definitional component' supplies the tools for representing the 'concepts', intended here as a formal representation of the 'important notions' of a given application domain. A concept is rendered as a frame-like data structure associated with a symbolic label like *human_being, taxi_* (the general class referring to all the possible taxis, not a specific cab), *city_, chair_, gold_*, or *escort_agency* in Table 1. These concepts are inserted into a generalization/specialization hierarchy that, for historical reasons, is called H_CLASS(es), and which corresponds well to the usual ontologies of concepts evoked in Section 1.

The 'enumerative component' concerns the formal representation, as (at least partially) instantiated frames, of the concrete realizations (lucy_, taxi_53, paris_, london_escort_agency) of the concepts. In NKRL, the formal representations of these instances take the name of *individuals*. Individuals are countable and, like the concepts, possess unique symbolic labels (lucy_ etc.). Throughout this paper, we will use the italic type style to represent a *concept_*, the roman style to represent an individual_.

The 'descriptive component' concerns the tools used to produce the formal representations (called 'templates' in the NKRL's parlance) of *general classes of narrative events*, like "moving a generic object", "formulate a need", "having a negative attitude towards someone", "be present somewhere", etc., see also Section 1 above. In contrast to the traditional *ternary* (name-attribute-value) frame-like structures used for concepts and individuals, templates are characterized by a *quaternary* format connecting together, essentially, the *symbolic name* of the template, a *predicate* (like BEHAVE, EXIST, PRODUCE…) and several *arguments* of the predicate. These last are, in turn, differentiated through the use of a set of named relations, the *roles* (like SUBJ(ect), OBJ(ect), SOURCE…). If we denote with

L_i the generic symbolic label identifying a given template, with P_j the predicate, with R_k the generic role and with a_k the corresponding argument, the template data structures have then the following format:

$$(L_i\ (P_j\ (R_1\ a_1)\ (R_2\ a_2)\ \ldots\ (R_n\ a_n)))\ . \qquad (1)$$

Templates are structured into an inheritance hierarchy, H_TEMP(lates), which corresponds to a new sort of ontology, an 'ontology of events'.

The instances (called 'predicative occurrences') of the templates, i.e., the representation of specific elementary events like "Tomorrow, I will move the wardrobe", "Lucy was looking for a taxi" or "The London Escort Agency provides an escort service" are in the domain of the last component, the factual one.

Returning now to the code of Table 1, we can say that, to represent a 'narrative' (in the most general meaning of this term) like "the London Escort Agency provides an escort service" under the form of a predicative occurrence (factual component), we must select firstly the template (descriptive component) corresponding to 'supply a service', which is represented in Table 2. This template is a specialization (see the 'father' code) of the particular PRODUCE template of H_TEMP corresponding to the 'production of immaterial entities'. In a template, the arguments of the predicate (the a_k terms in (1)) are represented by variables with associated constraints — which are expressed as concepts or combinations of concepts, i.e., using the terms of the H_CLASS hierarchy. The constituents (as SOURCE in Table 2) included in square brackets are optional; the constituents marked as '≠' are forbidden, see the BEN(e)F(iciary) role (a different template, of the MOVE type, is used in NKRL to represent the explicit transfer of a service to an individual or a social body).

When deriving a predicative occurrence, like c1 in Table 1, from a template, the role fillers in this occurrence must conform to the constraints of the father-template. For example, in occurrence c1, london_escort_agency is an individual, instance of the concept *company_* that is, in turn, a specialization of the concept *human_being_or_social_body*; escort_service_1 is an individual instance of *escort_service*, a specialization of *service_* (a specific term of *social_activity*), etc. Note that, in Table 1, the filler of the SUBJ(ect) role is a 'complex' one (expansion): the SPECIF(ication) operator is used here to introduce further details ("The London Escort Agency is a sort of United Kingdom escort agency") about the main constituent of the filler, see next Section. The 'location variables' like *var2* and *var5* in Table 2, and their corresponding instances, see london_uk in Table 1, are linked with the fillers (the arguments of the predicate) by using the colon (':') operator.

We can note a last, important point. The (about 200) templates that make up actually the H_TEMP hierarchy — the 'catalogue' of NKRL templates — are *permanent* and *fully defined*. We can say that these templates are part and parcel of the definition of the language. This approach is particularly advantageous for practical applications because it implies that: i) a system-builder does not have to create himself the structures needed to describe the events proper to a large class of Web narratives; ii) it becomes easier to secure the reproduction or the sharing of previous results. Moreover, when needed, it is easy to derive new templates from the existing ones. If they prove to be sufficiently general, they are then added to the 'catalogue'. H_TEMP is then a continuously growing structure.

Table2. Deriving an occurrence from a template

```
name : Produce:Services
father : Produce:ImmaterialEntities
position : 6.111
NL description : 'Production of Services'

PRODUCE     SUBJ        var1: [(var2)]
            OBJ         var3
            [SOURCE     var4: [(var5)]]
            ≠(BENF)
            [MODAL      var6]
            [TOPIC      var7]
            [CONTEXT    var8]
            { [ modulators ], ≠abs }

var1    =       <human_being_or_social_body>
var3    =       <service_>
var4    =       <human_being_or_social_body>
var6    =       <action_name>
var7    =       <sortal_concept>
var8    =       <event_> | <action_name>
var2, var5 =    <physical_location>
```

3 ADVANCED FEATURES OF THE NKRL LANGUAGE

The basic NKRL tools are enhanced by the use of two additional mechanisms:

- the AECS 'sub-language' [10] that allows the construction of complex (structured) predicate arguments called 'expansions';
- the second order tools (binding structures and completive construction) [11] used to represent the 'connectivity phenomena' (logico-semantic links) that, in a narrative situation, can exist between single narrative fragments.

Table 3 translates this fragment of Web news story: "This morning, the spokesman said in a newspaper interview that, yesterday, his company has bought three factories abroad". today_ and yesterday_ are two fictitious individuals introduced here, for simplicity's sake, in place of the real dates characterizing c2 and c3, see [11] on the NKRL coding of temporal information.

The operator SPECIF(ication) is one of the four operators that make up the AECS sub-language: the disjunctive (ALTERNative = A), distributive (ENUMeration = E), collective (COORDination = C), and attributive operator (SPECIFication = S).

The meaning of ALTERN is self-evident. The SPECIF lists, with syntax (SPECIF e_i p_1 … p_n), are used to represent some of the properties p_i that can be asserted about the first argument e_i, concept or individual, of the operator, e.g., human_being_1 and *spokesman_* in occurrence c2 of Table 3. In a COORD list, all the elements of the expansions take part — *necessarily together* — in the particular relationship with the predicate defined by the role to be filled. As an example, we can imagine a situation where the spokesman of Table 3 has transmitted his information to two different newspapers, newspaper_1 and newspaper_2, the BEN(e)F(iciaries). If the two newspapers have assisted

together to the interview, i.e., if they have received the information *together*, then the BENF slot of c2 in Table 3 will be filled with: (COORD newspaper_1 newspaper_2). On the contrary, if the information were received *separately* — which corresponds, in practice, to a situation where the two newspapers have taken part in two different interviews — then the BENF filler would have been: (ENUM newspaper_1 newspaper_2). The AECS operators and their arguments cannot be mixed together freely, see the 'priority rule' in [10].

Table3. An example of completive construction

```
c2)  MOVE   SUBJ  (SPECIF human_being_1 (SPECIF
                        spokesman_ company_1))
            OBJ   #c3
            BENF  newspaper_1
            MODAL interview_
            date-1: today_
            date-2:

c3)  PRODUCE
     SUBJ   company_1
     OBJ    (SPECIF purchase_1 (SPECIF factory_99
              (SPECIF cardinality_ 3))): (abroad_)
     date-1: yesterday_
     date-2:

[ factory_99
        InstanceOf:    factory_
        HasMember:     3 ]
```

The last element of Table 3 supplies an example of 'enumerative' data structure, see Section 2, explicitly associated with the individual factory_99 according to the rules for coding 'plural situations' in NKRL [10]. The non-empty HasMember slot in this structure makes it clear that the individual factory_99, as mentioned in c3, is referring in reality to several instances of *factory_*: in Table 3 we have supposed, in fact, that the three factories were not sufficiently important in the context of the story to justify their explicit representation as specific individuals.

In coding narrative information, one of the most difficult problems consists in being able to deal with the 'connectivity phenomena' like causality, goal, indirect speech, co-ordination and subordination, etc. — in short, all those phenomena that, in a sequence of statements, cause the global meaning to go beyond the simple addition of the information conveyed by each single statement. In NKRL, this is dealt with using second order structures obtained through a sort of *reification* of the predicative occurrences.

A very simple example of second order structure is given by the so-called 'completive construction' that consists in accepting as filler of a role in a predicative occurrence the symbolic label (reification) of another predicative occurrence. For example, the MOVE template at the origin of c2 in Table 3 is systematically used to translate any sort of explicit or implicit transmission of an information ("The spokesman said…"). In this example of completive construction, the filler of the OBJ(ect) slot in the occurrence (here, c2) which instantiates the 'transmission' template is a symbolic label (here, c3) that refers to the occurrence bearing the informational content to be spread out (" …the company has bought three factories abroad").

Table 4 corresponds now to a narrative information that can be rendered in natural language as: "We notice today, 10 June 1998, that British Telecom will offer its customers a pay-as-you-go (payg) Internet service".

To translate the general idea of 'acting to obtain a given result', we then use:

- A predicative occurrence (c5 in Table 4), instance of a template pertaining to the 'focusing on a result' sub-tree of the BEHAVE branch of H_TEMP. This occurrence is used to express the 'acting' component, i.e., it allows us to identify the SUBJ(ect) of the action, the temporal co-ordinates, possibly the MODAL(ity) or the instigator (SOURCE), etc.
- A second predicative occurrence, c6 in Table 4, which is used to express the 'intended result' component. This second occurrence, which happens 'in the future' with respect to the previous one (BEHAVE), is marked as hypothetical, i.e., it is always characterized by the presence of an uncertainty validity attribute, code '*'. Expressions like after-10-june-1998 are concretely rendered as date ranges, see [11].
- A 'binding occurrence', c4 in Table 4, linking together the previous occurrences and labeled with GOAL, an operator pertaining to the taxonomy of causality of NKRL [11].

Table 4. An example of binding occurrence

```
c4)  (GOAL  c5  c6)

c5)  BEHAVE  SUBJ  british_telecom
     { obs }
     date1:  10-june-1998
     date2:

*c6) MOVE    SUBJ  british_telecom
             OBJ   payg_internet_service_1
             BENF  (SPECIF customer_
                           british_telecom)
             date1: after-10-june-1998
             date2:
```

Binding structures — i.e., lists where the elements are symbolic labels, c5 and c6 in Table 4 — are then another example of second-order structures used to represent the connectivity phenomena. The general schema for coding the 'focusing on an intended result' domain is now:

c_α) (GOAL c_β c_γ)
c_β) BEHAVE SUBJ <*human_being_or_social_body*>
*c_γ) <predicative_occurrence, with any syntax>

In Table 4 'obs(erve)' is, like 'begin' and 'end', a temporal modulator, see [11]. 'obs' is used to assert that the event related in the occurrence 'holds' at the date associated with date-1 without, at this level, giving any detailed information about the beginning or end of this event, which normally extends beyond the given date. Note that the addition of a 'ment(al)' modulator in the BEHAVE occurrence, c_β, that introduces an 'acting to obtain a result' construction should imply that no concrete initiative is taken by the SUBJ of BEHAVE in order to fulfill the result. In this case, the 'result', *c_γ, reflects only the wishes and desires of the SUBJ(ect).

4 SOME REMARKS ON THE INFERENCE PROCEDURES

Search patterns are NKRL data structures that correspond to *partially instantiated templates* and that supply the general framework of information to be searched for, by filtering or unification, within an NKRL knowledge base — e.g., a knowledge base of Euforbia labels used for Web filtering.

The upper part of Table 5 is the representation of a very simple narrative fragment: "On June 12, 1997, John and Peter were admitted (*together* = `COORD`) to hospital". The 'temporal modulator' included in `c6`, `begin`, asserts that the date associated with `date-1` corresponds to the beginning of the state of being at the hospital. Modulators — deontic, modal (like `ment` in the previous Section), and temporal modulators like `begin, obs` and `end`) are special codes that are added to the basic core of a predicative occurrence to better specify its conceptual meaning, see [10]. A simple example of search pattern, translating the query: "Was John at the hospital in July/August 1997?" is then represented in the lower part of Table 5. The two timestamps associated with the pattern constitute now the 'search interval' that is used to limit the search for unification to the slice of time that it is considered appropriate to explore. In our example, this search pattern can successfully unify occurrence `c6` of Table 5: in the absence of explicit, negative evidence, a given situation is assumed to persist within the immediate temporal environment of the originating event, see [11].

Table 5. An example of search pattern

```
c6) EXIST   SUBJ  (COORD john_ peter_):(hospital_1)
            { begin }
            date-1: 2-june-1997
            date-2:

(?w     IS-PRED-OCCURRENCE
        :predicate     EXIST
        :SUBJ          john_
        :location of SUBJ      hospital_
        (1-july-1997, 31-august-1997))
```

In the Java, XML/RDF-compatible version of NKRL [15], a specific FUM (Filtering Unification) Module deals with search patterns. Unification is executed taking into account, amongst other things, the fact that a 'generic concept' included in the search pattern can unify one of its 'specific concepts' — or the instances (individuals) of a specific concept — included in a corresponding position of the occurrence. 'Generic' and 'specific' refer, obviously, to the structure of H_CLASS.

The inference level supplied by FUM is only a first step towards the set up of complex NKRL reasoning strategies, like 'transformations' and 'hypotheses', which require the use of inference engines having FUM as their inner core, see [14].

5 CONCLUSION

In this paper, we have introduced some properties of NKRL (Narrative Knowledge Representation Language), a conceptual modeling formalism used for high-level annotation purposes that takes into account, in particular, the semantic characteristics of those 'narratives' that represent a very large percentage of the global Web information. NKRL is characterized by the use of several representational principles (concepts under the form of frames, templates, second order binding structures etc.) and it implies the use of several high-level inference tools.

REFERENCES

[1] Kahan, J., Koivunen, M.-R., Prud'Hommeaux, E., and Swick, R., Annotea: An Open RDEF Infrastructure for Shared Web annotations. In: *Proc. of WWW10*. New York, ACM Press, 2001.

[2] Hillmann, D., *Using Dublin Core*. Dublin Core Metadata Initiative, 2001 (*http://dublincore.org/documents/2001/04/12/usageguide/*).

[3] *Dublin Core Qualifiers (DCMI Recommendation)*, 2000 (*http://dublincore.org/documents/dcmes-qualifiers/*).

[4] Zarri, G.P., Metadata, a 'Semantic' Approach. In: *Database and Expert Systems Applications – Proc. of DEXA'99*. Springer-Verlag, Berlin, 1999.

[5] Lassila, O., Swick, R.R., eds., *Resource Description Framework (RDF) Model and Syntax Specification*. W3C, 1999 (*http://www.w3.org/TR/REC-rdf-syntax/*).

[6] Brickley, D., Guha, R.V., eds., *Resource Description Framework (RDF) Schema Specification*. W3C, 1999 (*http://www.w3.org/TR/WD-rdf-schema/*).

[7] Hendler, J., and McGuinness, D.L., The DARPA Agent Markup Language, *IEEE Intelligent Systems* **15**(6) (2000) 72-73.

[8] Fensel, D., *et al.,* OIL in a Nutshell. In: *Knowledge Acquisition, Modeling, and Management – Proc. of the European Knowledge Acquisition Conference*, EKAW'2000. Springer-Verlag, Berlin, 2000.

[9] Heflin, J., Hendler, J., and Luke, S., Coping with Changing Ontologies in a Distributed Environment. In: *Proc. of the AAAI-99 Workshop on Ontology Management*. AAAI, Menlo Park (CA), 1999.

[10] Zarri, G.P., NKRL, a Knowledge Representation Tool for Encoding the 'Meaning' of Complex Narrative Texts, *Natural Language Engineering* **3** (1997) 231-253.

[11] Zarri, G.P., Representation of Temporal Knowledge in Events: The Formalism, and Its Potential for Legal Narratives, *Information & Communications Technology Law* **7** (1998) 213-241.

[12] Zarri, G.P., Knowledge Acquisition from Complex Narrative Texts Using the NKRL Technology. In: *Proc. of the 9th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*. Dept. of CS of the University, Calgary, 1995.

[13] McNaught, J., *et al.,* Integrated Document and Knowledge Management for the Knowledge-Based Enterprise. In: *Proc. of PAKeM 2000: Practical Applications of Knowledge Management Conference*. The Practical Applications Company, Blackpool, 2000.

[14] Zarri, G.P., A Knowledge Engineering Approach to Deal with 'Narrative' Multimedia Documents. In: *Perspective of System Informatics – Proc. of the Andrei Ershov 4th International Conference*. Springer-Verlag, Berlin, 2001.

[15] Zarri, G.P., A Conceptual Model for Capturing and Reusing Knowledge in Business-Oriented Domains. In: *Industrial Knowledge Management: A Micro-level Approach*. Springer-Verlag, London, 2000.