A semi-automatic system for conceptual annotation, its application to resource construction and evaluation

W.J. Black[1], J. M[c]Naught[1], G.P. Zarri[2], A. Persidis[3], A. Brasher[4], L. Gilardoni[5], E. Bertino[6], G. Semeraro[7], P. Leo[8]

[1]Department of Language Engineering, UMIST, Manchester, UK
{bill,jock}@ccl.umist.ac.uk
[2]Centre National de la Recherche Scientifique, Paris, France
[3]ASSETT–Biovista, Athens, Greece
[4]Pira International, Leatherhead, UK
[5]QUINARY SpA, Milano, Italy.
[6]Dipartimento di Scienze dell'Informazione, Università degli Studi, Milano, Italy
[7]Dipartimento di Informatica, Università di Bari, Italy
[8]Java Technology Center, IBM Semea Sud, Bari, Italy
http://concerto.ccl.umist.ac.uk/

Abstract

The CONCERTO project, primarily concerned with the annotation of texts for their conceptual content, combines automatic linguistic analysis with manual annotation to ensure the accuracy of fact extraction, and to encode content in a rich knowledge representation framework. The system provides annotation tools, automatic multi-level linguistic analysis modules, a partial parsing formalism with a more user friendly language than standard regular expression languages, XML-based document management, and a powerful knowledge representation and query facility. We describe the architecture and functionality of the system, and how it can be adapted for a range of resource construction tasks, and how the system can be configured to compute statistics on the accuracy of its automatic analysis components.

1. Background

The work described here is being carried out in the framework of the CONCERTO project (Esprit P29159, finishing in September 2000). Concerto is concerned with the conceptual indexing, querying and retrieval of digital documents (Bertino et al., 1999), i.e. textual documents stored in any kind of digital repository, from the WWW to digital libraries to corpora. The core activities of the project are to set up a full knowledge engineering software environment (KESE) to enable the computer-aided conceptual annotation of documents and to further enable intelligent information retrieval via these annotations. The *computer-aided* nature of the design of CONCERTO means that it is equally applicable to the construction and management of corpora, and it is this aspect of the potential of the system that we describe here.

CONCERTO can offer strong support to the organisation wishing to engage in conceptual corpus annotation. We fully exploit the possibilities of metadata as a vehicle to carry the conceptual annotations for a document. We do much more than simply tag documents with consistent classifications: this is accomplished in an early stage of CONCERTO, by the BSEE module. We further enrich documents by tagging them semi-automatically with metadata conceptual annotations that represent chunks of meaning relevant to the annotator. The result is a store of document-based knowledge that can be flexibly queried to return precise answers to queries, and that can be used for other knowledge-based purposes, in addition.

CONCERTO is primarily concerned with the evolution from information retrieval to fact retrieval and knowledge acquisition from textual sources. This is approached through the construction of an advanced prototype Knowledge Engineering Software Environment (KESE). The heart of the environment is a knowledge base management system in which narrative knowledge is expressed in a knowledge representation language (Zarri, 1992), super-imposed as annotations on the textual sources of that knowledge. The system allows knowledge-based retrieval of asserted and derived facts, supported by the evidence of the textual data from which it was derived.

The CONCERTO KESE provides a flexible architecture, in which it is possible to cater for a wide variety of document management tasks, including linguistic corpus management. Although that was not its primary objective, it is one of the ways in which it will be exploited by some of the partners. This is made possible by two key aspects of the design of the CONCERTO KESE. The first is that, recognizing the limitations of robust linguistic analysis, the system was always intended to rely on manual intervention to complete the annotation of textual data for storage in its conceptual knowledge base. The second is that the knowledge base management system is designed to cater for XML-encoded document annotations, in which the conceptual content of documents can be captured and queried. The two commercial pilot applications are primarily concerned with the semantic content of the documents, but within the project, linguistic analysis at lower levels precedes semantic annotation, and such annotations can also be stored in the system's repositories.

2. The CONCERTO Knowledge Engineering Software Environment

In the CONCERTO project, we are developing a new approach to the construction and management of knowledge repositories, based on *textual annotation*. In other words, a marriage of knowledge base

management, document management, IR and language engineering technologies. In a traditional knowledge base, terminology and assertions are expressed in a formal knowledge representation language, which allows inferences to be drawn. In the CONCERTO knowledge engineering software environment (KESE), such knowledge is represented as annotations to the textual sources from which the knowledge is acquired. We do not aspire to translate all the meaning of the texts to a knowledge representation language, only that which is relevant to the annotator's needs. The knowledge acquisition process in the CONCERTO KESE is semi-automatic, where the final authority for the annotations placed in the repository rests with the human annotator.

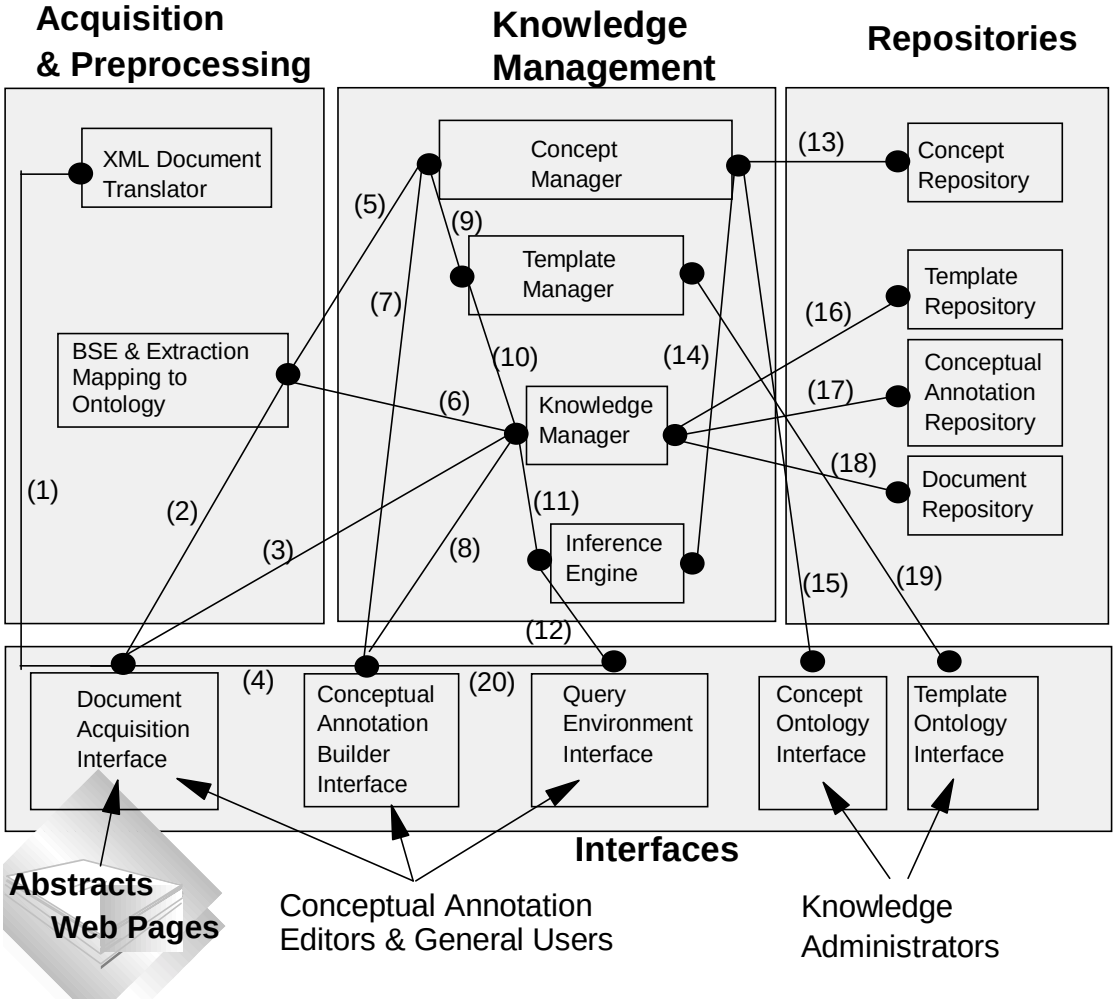

Figure 1 The Concerto system architecture

Attempting fully automatic knowledge acquisition from text is beyond the state of the art, as the results of MUC (1998) will testify. However, because such processes can deliver accuracy ((precision + recall)/2) over 90 percent for name extraction, and 80 percent for simple template filling, it is considered worthwhile to aid the annotator by pre-analyzing the text and proposing partially-filled annotations. The functionality of the generic KESE is shown in Figure 1. It includes the following processes:

1. Document capture and normalization to XML markup (Simpson, 1999).
2. Automatic low-level linguistic processing to identify names: companies, people, products, trade names, etc., to identify simple relationships between them and actions undertaken by them.
3. A process of conceptual annotation building, done interactively, in which a trained user corrects and augments the system's annotations. The interface aids this user in building representations which map natural language terms, names and phrases used in the text to object and template instances in the knowledge representation language NKRL (Zarri, 1997).
4. A knowledge management facility in which annotations expressed in NKRL are both stored as part of an XML document representation and indexed to support queries.
5. Facilities for querying the annotation knowledge base, and for generating reports and enhanced text to meet the users' requirements.

2.1. Document management

The *XML Document Translator* (XMLdt) is a Concerto KESE module used to encode, in a uniform

way, the documents input to the system. In the pilot application, two types of document are treated: abstracts of technical papers and Web pages containing company news and profiles.

Where a corpus is concerned, definition and creation of suitable XML Document Type Definitions (DTDs), would be required for the modelling of those kinds of documents that the Concerto KESE should manipulate. That is, the XMLdt module could be used to prepare an intial collection of documents in order to convert them to a common corpus format, or it could be used to process a corpus collection already in some format.

The XMLdt module inputs documents from the *Document Acquisition Interface* (DAI), outputting corresponding XML documents. These are sent via the DAI module to the *Knowledge Manager* (KM) for storage in the Conceptual Annotation repository.

2.2. Automatic text processing

Three components contribute to the construction of conceptual annotations in the CONCERTO KESE. The *Basic Semantic Element Extraction* (BSEE) module carries out linguistic processing to identify named entities and relationships between such entities as expressed directly in the text. The *Ontology Mapper* associates names, terms and partially filled templates with the classes and templates in the main KESE repositories, and the *Conceptual Annotation Builder Interface* (CABI) enlists the help of a user to complete the partial analysis achieved by the automatic analysis components. The BSEE and Ontology Mapper modules were originally developed within the scope of the FACILE project (Ciravegna et al., 1999), but their adaptation to CONCERTO requirements and the CABI module are new in CONCERTO.

2.2.1. Basic Semantic Elements Extractor

The BSEE module's purpose is to extract and tag basic semantic elements in a text, automatically. The BSEE module has been shown in blind tests to be capable of an accuracy (precision and recall) of over 90 percent at correctly tagging names in news text. We expect to be able to attain similar levels of accuracy in the CONCERTO applications, but 90% is not 100%, so the module's output needs to be checked by the CABI user after the tags are mapped to the CONCERTO ontology. We expect nonetheless that, despite the need for verifying results, the system will contribute to faster annotation than could be achieved in a purely manual system.

The BSEE module uses a modular language engineering solution to the problem of named-entity recognition, and extends the concept to the extraction of other basic semantic elements of relevance to the application domain. The strategy adopted is as follows:

The input text is tokenised (split into individual words, numbers, punctuation marks, and other symbols).

The tokenised text is morphologically analyzed and tagged, using components under licence from InXight.

Words and phrases are looked up in a database of known names, part names and headwords of phrases designating domain terms.

Finally, the *NE Analyser* (Black, Rinaldi and Mowatt 1988), using context-sensitive regular expression rules, augmented by Prolog-style unification of variables, builds up syntactic and semantic structural representations, and also finds instances of co-reference between names.

The FACILE project was mainly concerned with financial news (and has been successfully deployed in the marketplace), however the system and resources are not domain specific. Extensions to the resources are required only if new types of entities have to be captured. Thus, we have extended it in conjunction with our user partners to handle the entities of interest in printing and publishing, and biotechnology. The architecture itself is language-neutral and resources have been developed for four European languages. Recent extensions include a Java-based client that allows access to the system via a conventional web browser (Rinaldi and Black, 1999).

2.2.2. Ontology Mapper

The Ontology Mapper is developed from the QClassifier module, developed by Quinary for the FACILE system. Its task there was to use the terminology in the text to classify the topic of a text according to a fine-grained hierarchical ontology. In the CONCERTO project, the coverage is different, but also its usage is moer extensive. An ontology browser is an integral component of the annotation interface (see below) that permits users to compose conceptually more complete annotations than the BSEE module can make, and correct erroneous annotations. Although the conceptual annotations made in the pilot applications only call for selective fact extraction from input texts, this is not a restriction imposed by the system architecture. The linkage to ontology means that those text spans that are analyzed have their content words conceptually disambiguated, hence the OM combined with its annotator interface provides the generic functionality needed for intelligent lexical semantic tagging.

2.2.3. Conceptual Annotation Builder

Figure 2 CABI annotation interface

The Conceptual Annotation Builder (CAB), and its associated user interface CABI, allow the user to construct, using the results of the BSEE and OM as raw material, annotations representing the semantic content of text fragments. Annotators may work from scratch, if they prefer, although the results of the foregoing two modules make the process much more efficient.

The first prototype of the CABI annotation tool, is illustrated in Figure 2. The text pane shown to the left is a styled document that has already been automatically analyzed at a variety of levels by the BSEE and OM modules, but where the user can revise and extend the annotations. In the panel to the top right, there are instances of the objects as found either by automatic analysis or by manual annotation. This component is the template editor.

This early version used the ready-made Java *JTreeViewer* to present and edit representations of structured objects, with attributes and values shown as sub-trees. Work is ongoing at implementing a more ergonomic template viewer and editor that can do the same job more efficiently, and with a more natural presentation.

To the bottom right is a knowledge-base browser that can be accessed to select and apply annotations that the automatic analysis has not produced, or has produced in error. The knowledge base browser is focused on concepts instantiated by items extracted from the text, but also enables the annotator to search for missed items. Similarly, the template editor is focused on templates triggered in the text, with slots pre-filled with arguments picked up by automatic 'template relation' extraction.

2.3. Knowledge representation and NKRL

Conceptual annotations have been introduced for describing in some depth the context of documents and in general of multimedia objects (Catarci et al., 1997; Hoppe et al., 1996; Levy et al., 1996; Kirk, 1996; Welty, 1994). Unfortunately, most of the proposed approaches, often based on description logic, have several limitations in terms of description of complex events. They are often not adequate to describe the actions, facts, events, states, etc., that relate the real or intended behaviour of some actors, typical of any industrial and economic context. The *Narrative Knowledge Representation Language* (NKRL) we use as the basic language for indexing digital documents overcomes most of these limitations (Zarri. 1995; Zarri, 1997).

NKRL has been extensively described in the literature (see, e.g., Zarri (1992), Zarri (1995), Zarri (1997) and Zarri & Gilardoni (1996). It is a high-level knowledge representation language endowed with particular features which make it highly suitable for representing descriptive meanings like those proper to conceptual annotations. The core of NKRL consists of a set of general representation tools that are structured into four integrated components, that we discuss here in pairwise fashion:

2.3.1. Definitional and enumerative components

The definitional component of NKRL supplies the tools for representing the important notions (concepts) of a given domain; in NKRL, a concept is, therefore, a definitional data structure associated with a symbolic label like *physical_entity*, *human_being*, *city_*, etc. NKRL concepts are placed in a generalisation and specialisation hierarchy called, for historical reasons, H_CLASS(es), corresponding to the usual ontology of terms. The *enumerative component* of NKRL represents instances (lucy_, wardrobe_1, taxi_53, paris_) of the concepts of the H_CLASS. In NKRL, their formal representations take the name of *individuals*.

Concepts and individuals are represented as frame-based structures composed of an object identifier (OID) and of a set of characteristic features (slots) (Zarri, 1997). In the overall architecture, general concepts that belong to the upper levels of H_CLASS are represented in a catalogue and are assumed invariant across domains.

2.3.2. Descriptive and factual components

The event types in a domain, are represented using the *descriptive* and *factual* components. The *descriptive component* produces the formal representations (predicative templates) of general classes of narrative events, like 'move a generic object', 'formulate a need', 'be present somewhere'. In contrast to the binary structures used for concepts and individuals, templates have a threefold format where the central piece is a predicate, i.e., a named relation that exists among one or more arguments introduced by means of roles. The format of a predicative template is the following:

$(P_i\ (R_1\ a_1)\ (R_2\ a_2)\ ...\ (R_n\ a_n))$

In the above expression, P_i denotes the symbolic label identifying the template (class of events); R_k, $k=1,...,n$, denote generic roles; and a_k, $k=1,...,n$, denote the arguments associated with the roles. The predicates belong to the set {BEHAVE, EXIST, EXPERIENCE, MOVE, OWN, PRODUCE, RECEIVE}, and the roles to the set {SUBJ(ect), OBJ(ect), SOURCE, DEST(ination), MODAL(ity), TOPIC, CONTEXT}. Templates are structured into an inheritance hierarchy, H_TEMP(lates), which corresponds, therefore, to a taxonomy (ontology) of events. Instances (predicative occurrences) of the predicative templates, belong in the *factual component*. Predicative occurrences can be combined together, through the use of operators such as COORD, to form more complex annotations, called *binding occurrences*, the arguments of which can be either predicative or binding occurrences. The basic templates number more than 150, pertaining mainly to a (very broad) socioeconomic-political context where the main characters are human beings or social bodies. The basic templates can be specialized to derive templates for different applications.

Typically, each document is assigned a single *conceptual annotation*, representing the external binding occurrence associated with it.

As an example of a NKRL conceptual annotation, consider the following fragment: "On June 9, 1998, America Online announced it has finalised the acquisition of Mirabilis, an Israeli software house, for $287M cash. Mirabilis makes ICQ, an Internet tool enabling users to communicate with one another in real time". The conceptual annotation constructed from this text is shown in figure 3. The annotation is composed of a binding occurrence, c1, and three predicative occurrences, i.e., instances of basic templates included

in the catalogue. The second-order binding structure c1 analyzes the content into two main parts: an occurrence c2 relating the message, c4, transmitted by America Online (completive construction); and an occurrence, c3, describing ICQ. c3 and c4 are both instances of basic NKRL templates pertaining to the PRODUCE branch of the hierarchy of events. The presence of a temporal modulator, obs(erve), leads to an interpretation of c3 as the description of a situation that, at this particular date, is observed to exist. A location attribute (a list that contains here only one element) is associated with the SUBJ(ect) arguments in c2, c3 and c4. The arguments introduced by the OBJ(ect) and MODAL(ity) roles include some SPECIF(ication) lists, used to represent properties that can be asserted about the first element e_1, of the list. See Zarri (1997) for additional details.

```
c1) (COORD c2 c3)
c2) MOVE      SUBJ  america_online: (usa_)
              OBJ   #c4
              date-1: before-9-june-98
              date-2:
c4) PRODUCE   SUBJ america_online: (usa_)
              OBJ  (SPECIF purchase_1 final_(SPECIF mirabilis_
                        (SPECIF software_house israeli_)))
              MODAL (SPECIF cash_transaction_1  usa_dollar
                   (SPECIF amount_ (SPECIF million_ 287)))
              date-1: before-9-june-98
              date-2:
c3) PRODUCE   SUBJ mirabilis_: (israeli_)
              OBJ  (SPECIF icq_ (SPECIF internet_tool (SPECIF
                   communication_tool on_line)))
              [ obs ]
              date-1: 9-june-98
              date-2:
```

Figure 3: An example of a NKRL conceptual annotation

Conceptual annotations can be queried using *search patterns*, which are NKRL structures that can match, by filtering or unification, assertions in the conceptual annotation repository. Figure 4 shows an example NKRL search pattern meaning: "Who has recently acquired an Israeli software house, and according to which modalities?" that unifies with occurrence c4 in figure 3.

```
((?w   IS-OCCURRENCE
       :pred.   PRODUCE
       :SUBJ    ?x
       :OBJ     (SPECIF purchase_ (SPECIF ?y (SPECIF
                          software_house israeli_)))
       :MODAL ?z)
       (1_may_98, 31_july_98)
       ((?x     IS-A     (:OR human_being social_body))
                (?y      IS-A company_)
                (?z      IS-A general_sale_purchase_procedures)))
```

Figure 4: A simple NKRLsearch pattern

In figure 4, the two dates constitute the search interval associated with the search pattern: this interval is used to limit the search for unification to the slice of time that it is considered appropriate to explore, see Zarri (1998). Note that the concept `purchase_` behaves here as an implicit variable.

2.4. Knowledge management

In order to deal with NKRL conceptual annotations, information related to concept and template ontologies as well as information about conceptual annotations and documents is stored in secondary storage in appropriate repositories. Four repositories are used:

1. *Concept Repository*, storing concepts and instances belonging to the H_CLASS.
2. *Template Repository*, storing templates belonging to the H_TEMP, together with the information required to construct predicative occurrences starting from the considered templates.
3. *Document Repository*, storing the documents from which conceptual annotations have to be constructed and to which is associated the —
4. *Conceptual Annotation Repository*, storing conceptual annotations (in terms of predicative and binding occurrences) constructed starting from a set of documents.

Data in different repositories are related together, each document associated with one conceptual annotation comprising several predicative and binding occurrences. Each predicative occurrence instantiates one template referring to several concepts and instances.

The template, conceptual annotation and document repositories are also related by the fact that they all represent textual information: the template text, in the case of the template repository, the predicative or binding occurrences in the case of the conceptual annotation repository, and the documents themselves in the case of the document repository. To provide a homogeneous representation of all these types of information, we use the standard language for data exchange XML (Bradley, 1998). We represent templates and occurrences in Resource Description Format (RDF) (Lassila & Swick, 1999) and documents in XML. RDF is a proposal for defining and processing metadata developed by a W3C Working Group. The model, implemented in XML, makes use of directed labelled graphs and can be seen as a general tool box for implementing a specific (semantic) vocabulary, e.g. NKRL. Recent development of RDF already offers a basis for representing some of the most complex data structures present in NKRL (see for example the construct SPECIF used in figure 3). RDF higher order-statements can also represent the second-order structures of NKRL, like binding occurrences.

RDF is used to represent templates in the H_TEMP ontology, describing all their formal properties. This information is accessed by the CABI to express its output in RDF format for storage in the repository.

The repositories have been implemented in IBM's DB2 Universal Database, using the DB2 XML extender, recently released by IBM in beta version.

3. Conceptual annotation of corpora and the corpus annotator

In the previous section, we have concentrated on presenting the main components of CONCERTO. A few remarks have been made about the applicability of parts of the system to the needs of corpus annotators. We now turn to this application exclusively.

Corpus linguistics is now recognized as a core part of any language processing strategy. There is growing demand for large-scale language resources at many levels of linguistic description. Consequently, corpus builders demand tools that will help them carry out annotation tasks effectively. The CONCERTO system includes all the key elements needed for corpus construction, although these were not initially specialized for that task. These include:

1. Configurable tools for interactive annotation of text spans, including structured annotations.
2. Facilities for document and corpus management that scale up to managing large corpora.
3. The use of standard XML representations for exchange of information between processing modules and repositories.
4. High performance components for automatic analysis of texts at lower linguistic levels and partial user-specifiable analysis at intermediate and higher levels.
5. The ability to assert and and query complex annotations.

The "intelligent assistant" approach adopted by CONCERTO is one that recognizes the importance of such tasks but also the limitations of current technology, leaves the human in the driving seat and attempts to complement human strengths while overcoming human shortcomings.

3.1. Annotation user interfaces

For several levels of linguistic analysis, texts may be analyzed into non-overlapping text spans. Figure 2 illustrated this applied to named entities and domain terms. The Conceptual Annotation Repository maintains such representations internally, and can import and export them using XML encoding.

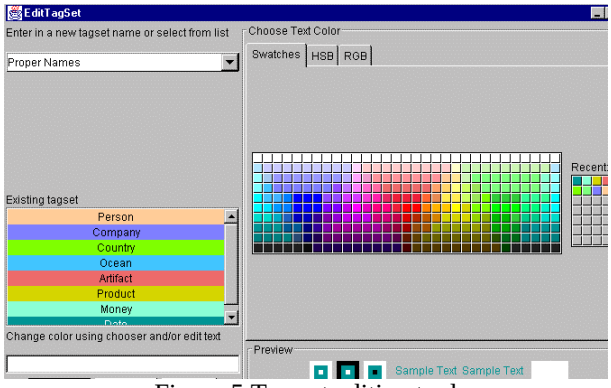

Figure 5 Tag set editing tool

The same type of markup rendering can be used to represent text elements such as part of speech tags, sense categories, noun phrase and sentence boundaries, topic boundaries, etc. Such an annotation interface, once developed, is easily configured, using the editing tool depicted in Figure 5, based on the standard Java Swing Color Chooser.

3.1.1. Building structured annotations

The automatic analysis tools can also fill *templates* representing application-significant semantic content, labeling the semantic relations between extracted names and terms and predicates. For this sort of annotation, the highlighting method of visualization is not sufficient, and a more elaborate interface is required.

The semantic templates constructed using CABI are typical of a range of structured representations of linguistic objects. Hierarchical representations have wide conventional currency, and can represent DAGs as well as trees, by co-indexing components. The early prototype of Concerto uses a simple expandable tree viewer for this kind of annotation. However, an editable template browser under construction will make editing easier for the user. This too will be configurable, allowing the corpus designer to present the template to both the annotator and end user in a way that hides unnecessary detail of the representation language used.

The level of automatic assistance that the annotator is given will depend on the domain specific resources developed within the system. One such resource that will be developed by the Knowledge Administrators (see Figure 1), and equally applicable to a corpus designer, is a set of *user templates*. These are application domain specific templates, which will be created to represent relationships between concept instances that are of importance to the application domain. The aim is to ensure the annotator has to write as little 'raw' NKRL as possible.

The resulting system is one which bears some similarities to the Alembic Annotator's Workshop (Day, 1999), which is designed to facilitate the creation of marked-up corpora for the evaluation of information extraction systems, and to bootstrap the system's resources using machine learning techniques. In comparison with Alembic, the CONCERTO KESE is potentially broader in application, due to the ability to annotate in a richer knowledge representation language than MUC-style templates, and in the support of advanced querying of the text base through its inference engine.

3.2. Large scale corpus management

It is already commonplace to use *either* the XML markup language *or* a database to manage a corpus. The CONCERTO KESE does both, and at the same time enables text fragments to be retrieved which match a query in conceptual content, providing a much more powerful corpus research tool than pattern-matching. Although it is not a requirement in the Concerto pilot applications to maintain the intermediate level linguistic annotations of texts, the repository management modules need no further development to make this possible.

3.3. Linguistic analysis components

The Concerto architecture is not intended to be a 'plug and play' generic linguistic processing framework like GATE (Cunningham et al 1997), and does not in its present form support alternative components for linguistic analysis. The tokenization, tagging and morphological analysis components are all from InXight. As well as accurately analyzing the news domain texts we have tested them on, these tools have the merit of being extremely fast in operation.

The BSEE analyser (Black, Rinaldi and Mowatt, 1998) uses a pattern matching language that was originally intended only to tag occurrences of complex proper names, dates and number expressions, as in the

MUC named entity task. It has turned out to be more flexible than needed for that level of analysis. Names are recognized using syntactic, semantic, and orthographic clues, together with local context. The rule formalism allows these attributes of each token and phrase to be tested, and arbitrary attribute values to be set via the unification of variables. There is also a co-reference mechanism, which applies to complex names and repetitions of parts of them. This rule language is more powerful for corpus pattern searching than regular expressions in a language like Perl, because of the ability to deal with several levels of analysis simultaneously. Syntactic patterns may be identified basing conditions on the output of the tagger, or alternatively, semantic patterns may be defined on the basis of the database of part names and clue phrases together with orthography. With the linkage to ontology in the OM, it is also possible to search for instances of patterns in a given semantic field. Thus, the system supports two main ways to interrogate corpora: the conceptual query interface designed for the demonstrator applications, and the linguistic pattern-matching language for matches at a surface level.

3.4. Automating evaluation

In using CABI, an annotator starts from analysis by the automatic linguistic analysis components just described. Each time a tagged item's tag is changed by the user is an error of the automatic analysis system. A tag removed is *spurious*, in the terminology used by MUC, and a tag added is one missed. By simply counting these changes and the number of items found automatically, the system can compute its error rate for every document processed. Similarly, inter-annotator agreement can be measured by registering the items annotated before and after a second or subsequent annotator marks up the same text. In this way, a more lightweight evaluation of a system's performance can be conducted than in formal competitive evaluations. This is more suited to the needs of projects to undertake glass-box evaluations of systems under development.

4. Conclusion

The CONCERTO conceptual annotation approach supports the widely-held corpus annotation view that something, no matter how partial, is better than nothing. That is, it allows annotators to rapidly build up partial conceptual descriptions of core relevant events associated with a document. It is more suitable for the annotation of domain specific (sub)corpora than for general corpora, given the need for domain specific ontologies and the relative tractability of processing domain specific text at the conceptual level rather than general language text. However, many language technology applications target specific domains, thus require annotated domain specific corpora. Our approach is however general in its application to several annotation purposes, i.e. from the original purpose of annotating abstracts (Pira) and corporate intelligence (Biovista) through the spectrum to the large-scale annotation of domain specific corpora.

Further information on CONCERTO [1] may be obtained at: http://concerto.ccl.umist.ac.uk/.

5. References

Bertino, E., Black, B., Brasher, A., Candela, V., Catania, B., Deavin, D., Esposito, F., McNaught, J., Persidis, A., Rinaldi, F., Semeraro, G. & Zarri, G-P. (1999) Concerto: Conceptual indexing, querying and retrieval of digital documents. *Proc. ICMCS'99 Special Event* "The European Community Day", June 1999, Florence, Italy, 1106–1109.

Black, W.J., Rinaldi, F. & Mowatt, D. (1998) FACILE: Description of the NE system used for MUC–7. In MUC (1998).

Bradley N. (1998) *The XML companion*. Addison-Wesley, Reading, Mass.

Catarci, T., Iocchi, L., Nardi, D. & Santucci, G. (1997) Conceptual views over the Web: Intelligent access to heterogeneous information. *Proc. 4th Knowledge Representation Meets Databases (KRDB) Workshop*.

Ciravegna, F., Lavelli, A., Mana, N., Matiasek, J., Gilardoni, L., Mazza, S., Ferraro, M., Black B., Rinaldi F. & Mowatt, D. (1999) FACILE: Classifying texts with pattern matching and IE. *Proc. IJCAI'99*, August 3–6, 1999, Stockholm, Sweden. 890–895.

Cunningham, H., Humphreys, K, Wilks, Y and Gaizauskas, R. Software Infrastructure for Natural Language Processing. *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP-97)*, Washington, April 1997.

Davenport, T.H. & Prusak, L. (1998) *Working knowledge: How organizations manage what they know*. Harvard Business School Press, Boston, Mass.

David Day, John Aberdeen, Lynette Hirschman, Robyn Kozierok, Patricia Robinson and Marc Vilain (1997) Mixed-Initiative Development of Language Processing Systems In: Fifth Conference on Applied Natural Language Processing, 1997, Association for Computational Linguistics, 31 March -- 3 April, Washington D.C

Hoppe, T., Kindermann, C., Paulus, K., Tolksdorf, R., Buu, E., Heimann, S., Schmiedel, A. & Volle, P. (1996) The MIHMA Project: A Web information service based on Description Logics. *Proc. WWW5 Workshop AI-based Tools to Help W3 Users*, INRIA, Rocquencourt.
http://www.info.unicaen.fr/~serge/3wia/workshop/

Kirk, T. (1996) Knowledge based access to information on the World Wide Web. *Proc. WWW5 Workshop AI-based Tools to Help W3 Users*, INRIA, Rocquencourt.
http://www.info.unicaen.fr/~serge/3wia/workshop/.

Lassila, O. & Swick., R. (1999) *Resource Description Framework (RDF) model and syntax specification*. Technical report, W3C.

Levy, A., Rajaraman, A. & Ordille, J. (1996) Querying heterogeneous information sources using sources descriptions. *Proc. 22nd Int. Conf. on Very Large Databases*. 251–262.

[1] We gratefully acknowledge the financial subvention of the European Union to the Concerto Project (ESPRIT, No. 29159).

MUC (1998) *Proceedings of the 7th Message Understanding Conference*. Fairfax, VA. Available at http://www.muc.saic.com/proceedings/muc_7_toc.html#infoextract.

Rinaldi, F & Black, B. (1999) A named entity extraction system and its Web extensions. *Proc. VEXTAL*, 22–24 November 1999, Venice, Italy, 197–202.

Simpson, J.E. (1999) *Just XML*. Prentice-Hall PTR, Upper Saddle River, NJ.

W3C (1999) *XML Schema*. Available at http://www.w3.org/TR/xmlschema-1.

Welty, C. (1994) Knowledge representation for intelligent information retrieval. *Proc. CAIA–94 Workshop on Intelligent Access to Digital Libraries.*

Zarri, G-P (1992) The descriptive component of a hybrid knowledge representation language. In Lehmann, F. & Rodin, E.Y (1992) *Semantic Networks in Artificial Intelligence*. Pergamon Press, Oxford.

Zarri, G-P. & Azzam, S. (1997) Building up and making use of corporate knowledge repositories. In Plaza, E. & Benjamins, R. (eds) (1997*) Knowledge acquisition, modeling and management*. Lecture Notes in Computer Science 1319. Springer, Berlin.

Zarri, G-P. (1995) Knowledge acquisition from complex narrative texts using the NKRL technology. *Proc. 9th Banff Knowledge Acquisition for Knowledge-based Systems Workshop*, Calgary.

Zarri, G-P. (1997) NKRL: A knowledge representation tool for encoding the meaning of complex narrative texts. *Natural Language Engineering* 3. Special Issue on Knowledge Representation for Natural Language Processing in Implemented Systems. 231–253.

Zarri, G-P. (1998) Representation of temporal knowledge in events: The formalism, and its potential for legal narratives. *Information & Communications Technology Law* 7. 213–241.

Zarri, G-P. and Gilardoni., L. (1996) Structuring and retrieval of the complex predicate arguments proper to the NKRL conceptual language. *Proc. Ninth Int. Symp. on Methodologies for Intelligent Systems*, Zakopane, Poland, 1996. 398–417.

Zarri, G-P., Bertino, E., Black, B., Brasher, A., Candela, V., Catania, B., Deavin, D., Di Pace, L., Esposito, F., Leo, P., McNaught, J., Persidis, A., Rinaldi, F., & Semeraro.,G. (1999) Concerto: An environment for the intelligent indexing, querying and retrieval of digital documents. In Raś, Z.W. & Skowron, A. (eds) *Foundations of Intelligent Systems*. Lecture Notes in Artificial Intelligence 1609. Springer, Berlin, 226–234.