From Controlled Document Authoring to Interactive Document Normalization

Aurélien Max
Groupe d'Étude pour la Traduction Automatique
GETA-CLIPS
Grenoble, France
aurelien.max@imag.fr

Abstract

This paper presents an approach to normalize documents in constrained domains. This approach reuses resources developed for controlled document authoring and is decomposed into three phases. First, candidate content representations for an input document are automatically built. Then, the content representation that best corresponds to the document according to an expert of the class of documents is identified. This content representation is finally used to generate the normalized version of the document. The current version of our prototype system is presented, and its limitations are discussed.

1 Document normalization

The authoring of documents in constrained domains and their translation into other languages is a very important activity in industrial settings. In some cases, the distinction between technical writers and technical translators has started to blur, so as to minimize the time and efforts needed to obtain multilingual documents. The paradigm of *translation for monolinguals* introduced by Kay in 1973 (Kay, 1997)[1] led the way to a new conception of the authoring task, which first materialized with systems involving human disambiguation (e.g. (Boitet, 1989; Somers et al., 1990)). A related paradigm emerged in the 90s (Hartley and Paris, 1997), whereby a technical author is responsible for providing the *content* of a document and a generation system produces multilingual versions of it. Updating documents is then done by updating the document content, and only some postediting may take place instead of full translation by a human translator.

Systems implementing this paradigm range from template-based multilingual document creation to systems presenting the user with the evolving text of the document (often called the *feedback* or *control* text) in her language, following from the WYSIWYM (What You See Is What You Meant) approach (Power and Scott, 1998).[2] *Anchors* (or *active zones*) in the text of the evolving document allow the user to specify further its semantics by making choices presented to her in her language. The underlying content representation is then used to generate the text of the document in as many languages as the system supports. The MDA (Multilingual Document Authoring) system (Dymetman et al., 2000; Brun et al., 2000) follows the WYSIWYM approach, but puts a strong emphasis on the well-formedness of document semantic content. More particularly, document content can be specified in terms of *communicative goals*, allowing the selection of messages which are contrastive within the modelled class of documents in no more steps than is needed to identify a predefined communicative goal. Figure 1 illustrates the architecture of a MDA system. A MDA grammar specifies the possible content representations of a document in terms of trees of typed semantic objects in a formalism inspired from Definite Clause Grammars (Pereira and Warren, 1980).

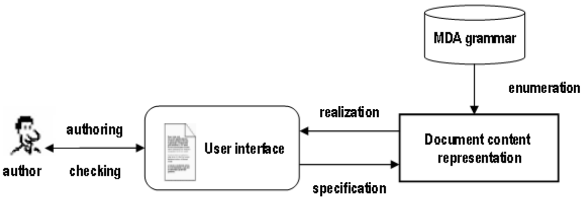


Figure 1: Architecture of a MDA system

[1] This is a reedition of the original article.

[2] We have done a review of these systems in (Max, 2003a) in which we have identified and compared five families of approaches.

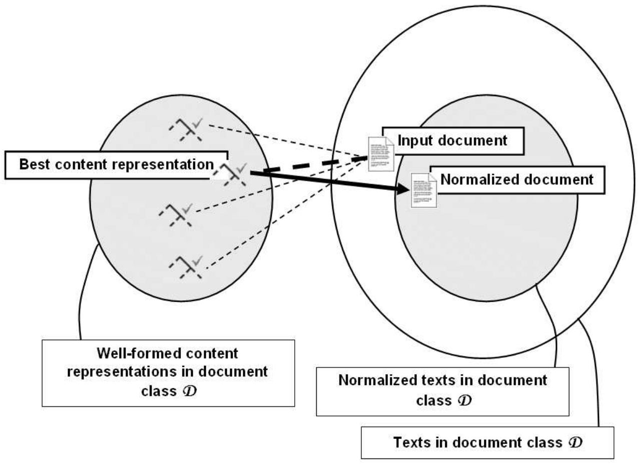

Best content representation
Input document
Normalized document
Well-formed content representations in document class 𝒟
Normalized texts in document class 𝒟
Texts in document class 𝒟

Figure 2: Document normalization in a given class of documents

Considering all the possibilities offered by having the semantic description of a document, for example the exploitation within the Semantic Web, it seemed very interesting to reuse resources developed for controlled document authoring to analyze existing documents. Also, a corpus study of drug leaflets that we conducted (Max, 2003a) showed that documents from the same class of documents could contain a lot of variation, which can hamper the reader's understanding. We defined *document normalization* as the process of the identification of the content representation produced by an existing document model corresponding best to an input document, followed by the automatic generation of the text of the normalized document from that content representation. This is illustrated in figure 2.

In the next section, we briefly describe our paradigm for document content analysis, which exploits the MDA formalism in a reverse way. Candidate content representations expressed in the MDA formalism are first produced and ranked automatically, and a human expert then identifies the one that best accounts for the communicative content of the original document. The core of this paper is devoted to our implementation of interactive negotiation for document normalization. Finally, we discuss our results and propose ways of improving the system.

2 Document normalization system

A MDA grammar can enumerate the well-formed content representations for documents of a given class and associate textual realizations to them (Dymetman et al., 2000). Content representations are typed abstract semantic trees in which dependencies can be established through unification of variables. Generation of text is done in a compositional manner from the semantic representation. Figure 3 shows an excerpt of a MDA grammar describing well-formed commands for the Unix shell. Such a grammar describes both the abstract semantic syntax and the concrete syntax for a particular language, English in this case. The first rule reads as follows: lsCommand is a semantic object of type shellCommand_type, which is composed of an object of type fileSelection_type, an object of type sortCriteria_type, and an object of type displayOptions_type. Text strings appearing in the right-hand side of the rules are used together with the strings associated with the present semantic objects to compose the normalized text associated with the described abstract semantic trees.

Our approach to normalize documents has been described in (Max, 2003b). A heuristic search procedure in the space of content representations defined by a MDA grammar is first performed. Its evaluation function measures a similarity score between the document to be normalized and the normalized documents that can be produced from a partial content representation. The similarity score is inspired from information retrieval, and takes into account common descriptors and their relative informativity in the class of documents. The *admissibility* property of the search procedure guarantees that the first complete content representation found is the one with the best global similarity score. This process uses text generation to measure some kind of similarity, and has been called *fuzzy inverted generation.* In order to better cover the space of texts conveying the same communicative content, the MDA formalism has been extended to support non-deterministic generation, allowing the production of competing texts from the same content representation, as is illustrated in figure 4. For each considered content representation, texts are produced and compared to the document to be normalized, thus allowing the ranking of candidate content representations

```
% semantic object of type 'shellCommand_type' describing the 'ls' command
lsCommand(FileSelection, SortCriteria, DisplayOptions)::shellCommand_type-e-[] -->
    ['List '],
    FileSelection::fileSelection_type-e-[],
    SortCriteria::sortCriteria_type-e-[],
    DisplayOptions::displayOptions_type-e-[].

fileSelection(ListOfFilesAndDirectories, HiddenFilesSelection, DirectoriesContentsListing, LinksReferenceListing)::
  fileSelection_type-e-[] -->
    ListOfFilesAndDirectories::listOfFilesAndDirectories_type-e-[], ['.'],
    HiddenFilesSelection::hiddenFilesSelection_type-e-[],
    DirectoriesContentsListing::directoriesContentsListing_type-e-[],
    LinksReferenceListing::linksReferencesListing_type-e-[].
% ...

% description for the type 'linksReferencesListing_type'
type_display(e, linksReferencesListing_type, 'specifies how links are shown').
% description for the objects 'displayLinksReferences' and 'dontDisplayLinksReferences'
functor_display(e, displayLinksReferences,'show the files and directories that are referenced by links').
functor_display(e, dontDisplayLinksReferences,'show links as such (not the files and directories they point to)').

displayLinksReferences::linksReferencesListing_type-e-[] -->
    [' Display referenced files and directories instead of links. '].
dontDisplayLinksReferences::linksReferencesListing_type-e-[] -->
    [' Display links as such. '].
```

Figure 3: MDA grammar extract for the description of the ls Unix command

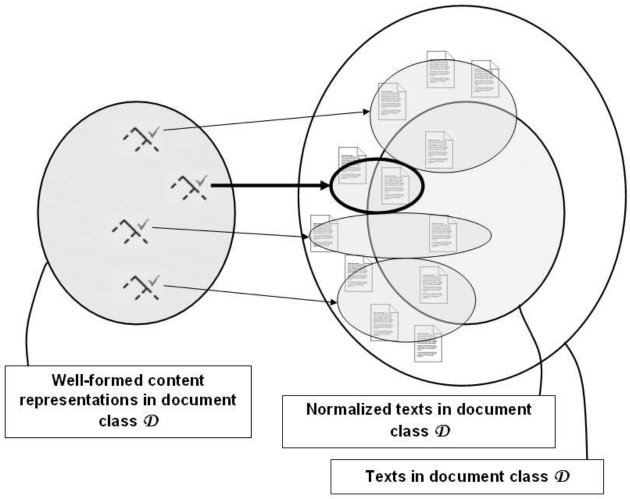

Well-formed content representations in document class 𝒟
Normalized texts in document class 𝒟
Texts in document class 𝒟

Figure 4: Identification of the best content representation through fuzzy inverted generation

by decreasing the similarity score.

Given the limitations of the similarity measure inspired from information retrieval, the search is continued to find the N first documents with the best similarity scores. The identification of the content representation that represents best the communicative content of the original document is then done by *interactive negotiation* between an expert of the class of the document and the system based on the candidates previously extracted.

To demonstrate how the implemented system works, we will consider the normalization of the following description in English of a command for the Unix shell with the grammar of figure 3: List all files. Do not show hidden files and visit subdirectories recursively. Sort results by date of last modification in long format in single-column in reverse chronological order. Give file size in bytes.

2.1 Finding candidate document representations: fuzzy inverted generation

The MDA grammar used is first precompiled offline by a separate tool, in order to associate profiles of text descriptors to semantic objects and types in the grammar (see (Max, 2003b) for details). In our current implementation, descriptors are WordNet synsets. The text of the input document is then lemmatized and the descriptors are extracted, yielding the profile of descriptors for the input document. The grammar is then used to construct partial abstract semantic trees, which are ordered in a list of candidates according to the similarity score computed between their profile and that of the input document. At each iteration, the search algorithm considers the most promising candidate content representation and performs one step of derivation on it, which corresponds to instantiating a variable in the tree with a value for its type. The first complete candidate (i.e. an abstract tree not containing any variable) found is then kept, and the search continues until a given number of candidates has been found. This number defines a value of *system confidence*, which can be selected by the user of our normalization system: the higher the confidence, the fewer candidates are kept, at the risk that the best one according to an expert may not be present. Given the size of the grammar used and the complexity of the analysed document, a small number of candidates can be kept (20 in our example).

This process restricts the search space from a large collection of virtual documents[3] to a comparatively smaller number of concrete textual documents, associated with their semantic structure. A factorization process then builds a unique content representation that contains all the different alternative subtrees found in the candidates. Each semantic object in the resulting factorized semantic tree is then decorated by a list of all the candidates to which it belongs. Competing semantic objects are ranked according to the score of the candidate with the highest score to which they belong. This compact representation permits to consider *underspecifications* from the analysis of the input document present at any depth in the candidate semantic trees.

2.2 Identifying the best document representation: interactive negotiation

Document normalization implies a normative view on the analysis of a document. Because the communicative content that will be ultimately retained may not be exactly that of the original document, some negotiation must take place to determine which alternative semantic content, if any, is acceptable. This is analoguous to what happens in translation. As (Kay et al., 1994) put it:

> Translation is not a meaning-preserving function from a source to a target text. Indeed, it is probably not helpful to think of it as a function at all, but rather as a matter of compromise.

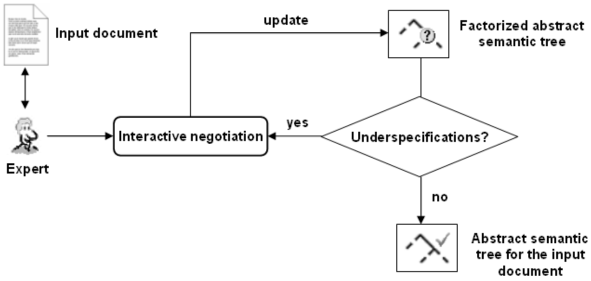

Input document
update
Factorized abstract semantic tree
Expert
Interactive negotiation
yes
Underspecifications?
no
Abstract semantic tree for the input document

Figure 5: Resolving underspecifications by interactive negotiation

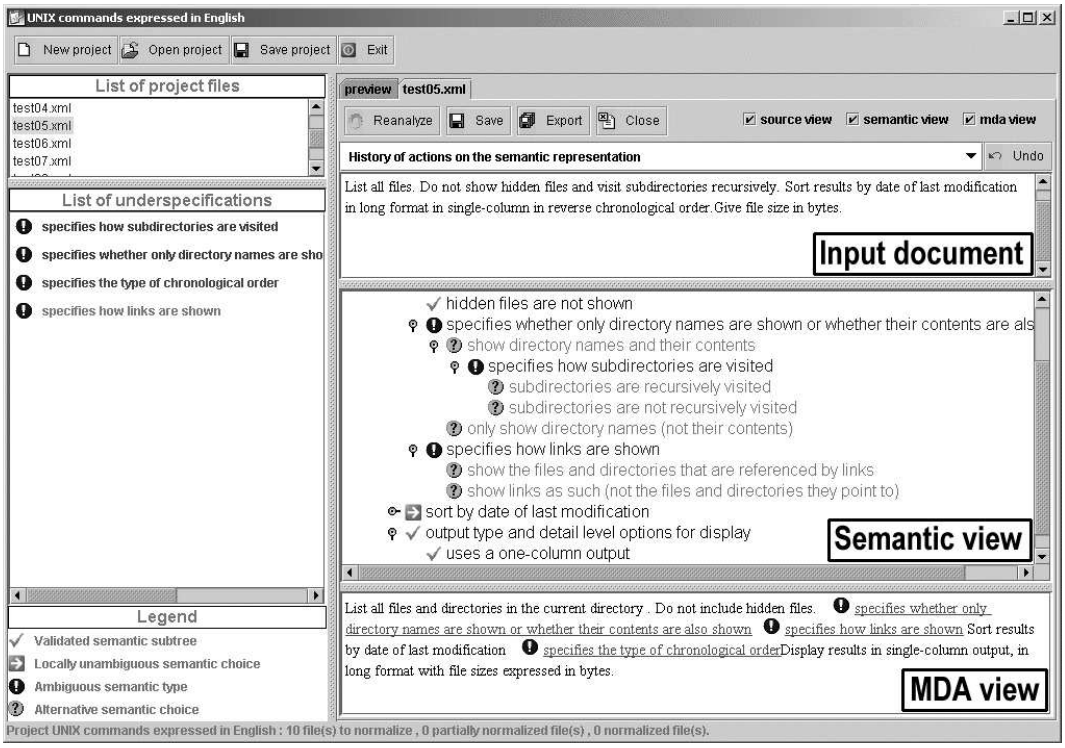
In our view, a human expert should be responsible for making difficult decisions that the machine cannot make without significant interpretation capabilities. Furthermore, these decisions encompass cases where no explicit content in the input document can be used to determine content that is expected in order to obtain a well-formed representation in the semantic model used.[4] This will be illustrated below with the negotiation dialogue of figure 8.

A naive way to select the candidate content representation found by the system that best corresponds to the input document would be to show to an expert all the normalized texts corresponding to the candidates. This would however be a tedious and error-prone task. The compact representation built at the end of fuzzy inverted generation allows the discrimination of candidates based on local underspecifications corresponding to competing semantic choices. We have implemented three methods for supporting interactive negotiation that will be described below. They allow an expert to resolve underspecifications and therefore update the factorized content representation by eliminating incorrect hypotheses. This is iterated until the factorized content representation does not contain any underspecification, as illustrated in figure 5.

Figure 6 shows the main interface of our normalization system after the automatic selection

[3] We call *virtual documents* all the documents that can be produced by a given grammar.

[4] This suggests that document normalization can be used as a corrective mecanism applied on ill-formed documents that can be incomplete or semantically incoherent relatively to a given semantic model.



❗ specifies how subdirectories are visited
❗ specifies whether only directory names are sho
❗ specifies the type of chronological order
❗ specifies how links are shown

List all files. Do not show hidden files and visit subdirectories recursively. Sort results by date of last modification in long format in single-column in reverse chronological order. Give file size in bytes.

```
✓ hidden files are not shown
❗ specifies whether only directory names are shown or whether their contents are als
  ❓ show directory names and their contents
    ❗ specifies how subdirectories are visited
      ❓ subdirectories are recursively visited
      ❓ subdirectories are not recursively visited
  ❓ only show directory names (not their contents)
❗ specifies how links are shown
  ❓ show the files and directories that are referenced by links
  ❓ show links as such (not the files and directories they point to)
➡ sort by date of last modification
✓ output type and detail level options for display
  ✓ uses a one-column output
```

List all files and directories in the current directory . Do not include hidden files. ❗ specifies whether only directory names are shown or whether their contents are also shown ❗ specifies how links are shown Sort results by date of last modification ❗ specifies the type of chronological orderDisplay results in single-column output, in long format with file sizes expressed in bytes.

✓ Validated semantic subtree
➡ Locally unambiguous semantic choice
❗ Ambiguous semantic type
❓ Alternative semantic choice

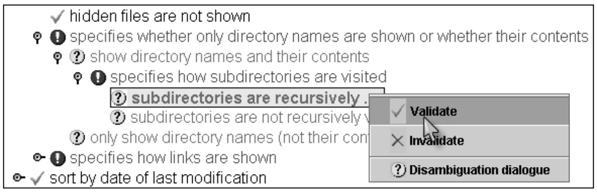
Project UNIX commands expressed in English : 10 file(s) to normalize , 0 partially normalized file(s) , 0 normalized file(s).

Figure 6: Interface of our document normalization system

of candidate content representations and the construction of the compact representation.

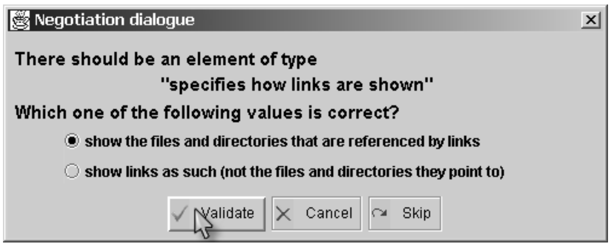
Semantic view The middle panel on the right of the window contains the *semantic view*, which is a graphical view of the factorized abstract semantic tree that can be interpreted by the expert. It uses the text descriptions for semantic objects and types as described by the functor_display and type_display predicates present in the original MDA formalism (see figure 3). The tick symbol ✓ represents a semantic object that dominates a semantic subtree containing no underspecifications. In our example, this is the case for the object described as output type and detail level for display. The arrow symbol ➡ describes a semantic object that does not take part in an underspecification, but which dominates a subtree that contains at least one. The exclamation mark symbol ❗ denotes a semantic type that is underspecified, and for which at least two semantic objects are in competition. Semantic objects in competition are denoted by the interrogation mark symbol ❓, and are ordered according to the highest score of the candidate representation to which they belong.

This view can be used by the expert to navigate at any depth inside the compact representation. By clicking on a semantic object in competition, the expert can decide whether this object belongs to the solution or not. On the example of figure 7, the expert has selected the first possibility (subdirectories are recursively visited) for an underspecified type (specifies how subdirectories are visited), which is itself dominated by another underspecified type (specifies whether only directory names are shown or. . .). The menu that pops up allows the *validation* of the selected object: this will have for effect to prune the factorized tree of any subtree that does not belong to at least one of the candidates of the validated object. In the present case, not only will it prune the alternative subtree dominated by subdirectories are not recursively visited, but also the subtree dominated by only show directory names (not



```
✓ hidden files are not shown
❗ specifies whether only directory names are shown or whether their contents
  ❓ show directory names and their contents
    ❗ specifies how subdirectories are visited
      ❓ subdirectories are recursively
      ❓ subdirectories are not recursively v
  ❓ only show directory names (not their con
❗ specifies how links are shown
✓ sort by date of last modification
```

Figure 7: Validation of a semantic choice within the semantic view


Negotiation dialogue

There should be an element of type

"specifies how links are shown"

Which one of the following values is correct?

● show the files and directories that are referenced by links

○ show links as such (not the files and directories they point to)

Figure 8: Negotiation dialogue about how links should be shown

their content) present at a shallower level in the tree. Furthermore, subtrees that would be incompatible elsewhere in the compact representation because of failed parameter unification would disappear. Conversely, the *invalidation* operation prunes all the subtrees which have at least one candidate in common with the invalidated object. The expert can also ask for a negotiation dialogue, which will be introduced shortly.

MDA view It seemed very natural to propose a view with which a user of a MDA system would already be familiar. Such a view shows the normalized text corresponding to all the objects from the root object that are not in competition. Underspecified semantic types appear as underlined text spans called *active zones*, which trigger a pop up menu when clicked. Whereas in the MDA authoring mode all the possible objects for the semantic type that do not violate any semantic dependencies are shown, our system only proposes those that belong to candidates that are still in competition. Furthermore, these semantic objects are not ordered by their order in appearance in the grammar, but by the score of their most likely candidate according to our system. Selecting an object corresponds to validating it, implying that the invalidation operation is not accessible from this view. Also, underspecified semantic types dominated by other underspecified types cannot be resolved using this view, as they do not appear in the text.[5] However, dealing with a text in natural language corresponding to the normalized document may be a more intuitive interface to some users, although it may require more operations.

Negotiation dialogues The key element in this task is the minimization of the number of operations by the user. The two previous views allow the expert to choose some underspecification to resolve. The List of underspecifications panel on the left of the window in figure 6 contains an enumeration of all underspecifications found in the compact representation. They are ordered by decreasing score, where the score can indicate the average score of the objects in competition, or the inverse of the average number of candidates per object in competition. Therefore, the expert can choose to resolve first underspecifications that contain likely objects, or underspecifications that involve few candidates so that the validation of an object will prune more candidates from the compact representations. Clicking on an underspecification in the list triggers a negotiation dialogue similar to that of figure 8. The semantic type on that dialogue, specifies how links are shown, is not supported by any evidence in the input document. The expert can however choose a value for it. When the underspecification is resolved, all the views are refreshed to reflect the new state of the compact representation, and a negotiation dialogue for the underspecification then ranked first in the list is shown. The expert can either discard it, or continue in the dialogue mode, with the possibility to skip the resolution of an underspecification.

3 Discussion and perspectives

We have presented an approach to normalize documents in constrained domains and its implementation. Our approach combines the strictness of well-formed content representations and and the flexibility of information retrieval techniques, and makes use of human expertise to resolve difficult interpretation problems in an attempt to build an operational system. Although our initial results are promising, our approach could be improved in several ways.

First of all, an important evaluation factor of our approach is how much effort has to be done by the human expert. We have only conducted informal experiments of evaluation by the task, which have revealed that normalization can be performed quite fast when the user has a good command of the different views available. Nevertheless, it seems crucial to be able to present the expert with at least some evidence

[5] We thought that showing these types using cascade menus would be too confusing for the user.

from the text of the input document to support competing semantic objects. Morever, the evidence extracted from the input document could be used as the basis for learning new formulations for particular communicative goals that would match better subsequent similar input. Although our system already supports non-deterministic generation, we have not implemented a mechanism that would allow supervised learning of new formulations yet. We expect this "normalization memory" functionality to have an important impact for the normalization of documents from the same origin, as it should improve the automatic selection of content representations.

In case the candidates returned by fuzzy inverted generation do not contain the content representation representing best the input document, the user can choose to reanalyze the document. This will start search again from the (N+1)th content representation. But because the expert might have already resolved some underspecifications and thus identified subparts that should belong to the solution, this information should be taken into account while reanalyzing the document, which is not the case in the current implementation. If the solution has to be present in the list of candidates returned, it should be as close to the top of the list as possible, so that the first choices for each underspecification represent the actual best choices. To this end, we intend to implement a second-pass analysis that would rerank the candidates produced by fuzzy inverted generation by computing text similarities over short passages such as those proposed in (Hatzivassiloglou et al., 1999). These techniques were much harder to implement during the search in the virtual space of documents produced by the document model, because partial content representations are not actual texts.

4 Acknowledgements

Many thanks to Marc Dymetman, who supervised my work at Xerox Research Centre Europe and who originally came up with the concept of fuzzy inverted generation. Many thanks also to Christian Boitet, my university PhD supervisor, and to Anne-Lise Bully, Cédric Leray and Abdelkhalek Rherad for their programming work on the interface of the presented system. This work was funded by a PhD grant from ANRT and XRCE.

References

Christian Boitet. 1989. Speech Synthesis and Dialogue Based Machine Translation. In *Proceedings of the ATR Symposium on Basic Research for Telephone Interpretation, Kyoto, Japan.*

Caroline Brun, Marc Dymetman, and Veronika Lux. 2000. Document Structure and Multilingual Authoring. In *Proceedings of INLG 2000, Mitzpe Ramon, Israel.*

Marc Dymetman, Veronika Lux, and Aarne Ranta. 2000. XML and Multilingual Document Authoring: Convergent Trends. In *Proceedings of COLING 2000, Saarbrucken, Germany.*

Anthony F. Hartley and Cécile L. Paris. 1997. Multilingual Document Production - From Support for Translating to Support for Authoring. *Machine Translation*, 12:109–128.

Vasileios Hatzivassiloglou, Judith L. Klavans, and Eleazar Eskin. 1999. Detecting Text Similarity over Short Passages: Exploring Linguistic Feature Combinations via Machine Learning. In *Proceedings of EMNLP/VLC-99, College Park, United States.*

Martin Kay, Jean Mark Gawron, and Peter Norvig. 1994. *Verbmobil – A Translation System for Face-to-Face Dialog.* CSLI Lecture Notes.

Martin Kay. 1997. The Proper Place of Men and Machines in Language Translation. *Machine Translation*, 12:3–23.

Aurélien Max. 2003a. *De la création de documents normalisés à la normalisation de documents en domaine contraint.* PhD thesis, Université Joseph Fourier, Grenoble.

Aurélien Max. 2003b. Reversing Controlled Document Authoring to Normalize Documents. In *Proceedings of the EACL-03 Student Research Workshop, Budapest, Hungary.*

Fernando Pereira and David Warren. 1980. Definite Clauses for Language Analysis. *Artificial Intelligence*, 13.

Richard Power and Donia Scott. 1998. Multilingual Authoring using Feedback Texts. In *Proceedings of COLING/ACL-98, Montréal, Canada.*

H. Somers, J.-I. Tsujii, and D. Jones. 1990. Machine Translation without a Source Text. In *Proceedings of COLING-90, Helsinki, Finland*, volume 3, pages 217–276.