Document retro-conversion for personalized electronic reedition

A. Belaïd[1], A. Alusse[1], Y. Rangoni[1], H. Cecotti[1],
F. Farah[2], N. Gagean[2], D. Fiala[2], F. Rousselot[2], H. Vigne[3]

[1]LORIA, Campus Scientifique, B.P. 236, Vandoeuvre-Lès-Nancy, France
2LIIA INSA Strasbourg, Bd de la Victoire, 67000 Strasbourg
[3]EVER TEAM Company, 4pl Félix Eboué, Paris 12

Abstract

In this paper, we propose a generic framework to store, retrieve, transform and present mixed sets of native and virtual documents.
We intend to use or to develop specific tools organized in a global architecture, from document analysis and capture, document retrieval and classification-categorization, to full generation of personal sets of documents, corresponding to user's specific needs and profile.
The first step concerns document preparation and formal analysis. The second step adds semantic metadata, content indexing, and structure-semantic analysis. The third step helps user for the constitution of personalized documents.
Research is based on domain specific large sets of documents, as for example European Union law documents (many millions, many file formats, in twenty official languages).

INTRODUCTION

Nowadays, needs concerning the documentation have evolved from information constitution to its intelligent consultation. Indeed, the mass of documents related to the increasingly significant storage capacities, leads the information systems to introduce intelligence into the search processes and personalized exploitation, by taking more and more into account user requirements.

Project RNTL[1] PAPLOO positions in this area. It aims at the definition of a generic framework of transformation and document retrieval for personalized use. The user requirements are taken into account in all the chain processes. The first step of PAPLOO relates to the document preparation. It allows to dispose of documents in most detailed form

[1] Réseau National de Technologies Logicielles

as related to their structure and content. For slightly or not structured documents, the project considers further recognition steps and structure retro-conversion, followed by content indexing. Document retrieval is operated on the basis of keywords extracted from user requests. This search is then refined on the content in order to better satisfy the user needs in terms of structure and layout.

This approach belongs to the general research area related to Information Research System (IRS) where the objective remains the increasingly refined search for increasingly complex information in an increasingly significant documentation base. This triple complexity often obliges to combine several techniques to reach a better level of "perfection". Information retrieval can be operated according to case's at a global level of the document for the constitution of sub-databases or at its component level (fragments or sub-structures) for a finer information extraction, like in (Laine-Cruzel, 1999). Other works aim at, as in PAPLOO (Iksal, 2002), through their research, the constitution of personalized documents, called Personalized Virtual Documents (PVD) (Watters, 1999).

Finally, to answer the ergonomics constraints which most significant are the user feeling and the physical support constraints, part of the project relates to page-setting and to the typography which we will name re-formatting. Work on the adaptive ergonomics and hypermedia (Brusilovski, 1996) can be from this point of view connected to the work undertaken within the framework of PAPLOO. In this field, we are also interested in S. Ranwez research on the composition of the adaptive hypermedia (Ranwez, 2000) or in those mentioned in the workshop ICWE (ICWE, 2004).

The originality of PAPLOO lies especially in the fact that the purpose touches several fields and that several methods resulting from each field are combined in a coherent way to offer to a user a complete system which goes from the constitution of its database to the generation of a personal document. The chain consists of independent and plug-in modules.

SYSTEM OVERVIEW

Chain PAPLOO is composed of two distinct parts as shown in Fig.1. The first part relates to data preparation in terms of OCR, structure recognition and annotation. The second part concentrates on the constitution of files in terms of reformulation and reformatting. We will see these parts in more details later. The pivot language used throughout the chain is XML. Effective research starts after database constitution enriched by indices (metadata). The user query conditions all the chain. It initialises the total document research (classification, enrichment and reorganization) and allows the constitution of personalized files (documents) by providing the suitable elements of selection. In addition to the personal request, the influence of the user is present in a permanent way in all the phases of the system through his profile.

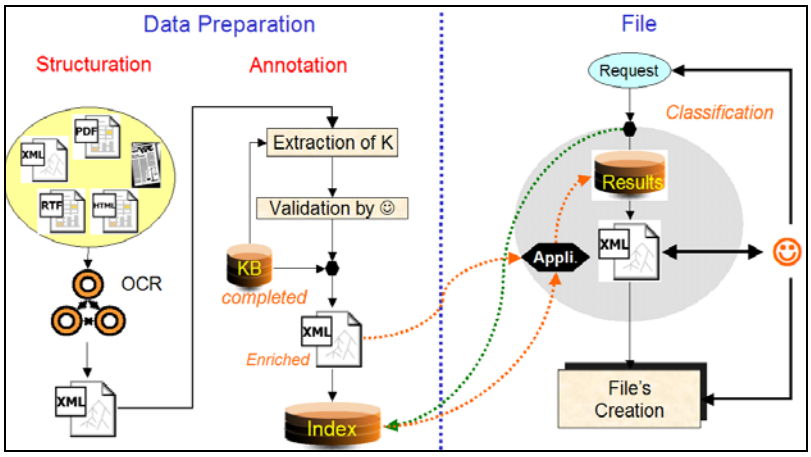

Fig.1 Chain PAPLOO overview

Documents used are law articles of all kinds belonging to Official Journals (OJ) of the European Union. They may be for example Council regulations (EC, EURATOM), Council decisions, Commission decisions, European Court Reports, Case-law, Judgment and opinions of the Court of Justice. Most of these documents appear in the EU Official Journal, every day in twenty official languages, with translations "cover-to-cover". It consists of two related series (L for legislation and C for information and notices) and a supplement (S for public procurement). There is also an electronic section to the C series, known as the OJ C E. Documents DBs consist in many millions of source documents (in TIFF, PDF, DOC, HTML, XML, TXT formats). Post-production processes are needed on these documents, as for example extracting a part from a PDF or TIFF page corresponding to a unit of information, logically separated from other logical documents present on this page.

TEXT RECOGNITION

Some of these documents are in PDF format and not directly readable. Their interrogation by the PAPLOO chain needs to convert them in a comprehensible data. However, as the characters are usually printed and of a good quality of printing, recognition has naturally called upon commercial OCRs. Based on this, we proposed a recognition procedure initiated by OCR combination, followed by an adaptive ICR for rejection processing.

OCR combination

Commercial OCRs are usually trained to recognize all kind of document and are not specialized for one of them particularly. As a consequence, this generic characteristic of OCRs ensures a good performance on the majority of characters but unavoidably leads, for a low proportion of them, to a bad performance, as they are less frequent or do not correspond to the character trained models. To overcome this drawback, the idea is to combine OCRs in

order to help each other mutually. This help will be more significant since the complementarity is large. Several combination methods are described in the literature: voting methods, Bayesian combination, Dempster-Shafer, behavior-knowledge space, neural networks, etc (Bahler 2000, Gunes 2004).

The voting method assumes that at less one classifier supports the character. The recognition is reached when all the classifiers support the character. This has as consequence to avoid correcting the common errors and leads to reject the maximum of errors instead of correcting them. Considering only two OCRs, the phenomena are accentuated, as there are less correction possibilities.

One of the strongest conditions to combine several classifiers based on conditional probability as formulated in the Bayes' rule is that the classifiers must act independently. This condition is not easy to verify. It's why it favors the use of the behavior-knowledge space (BKS) method (Suen 2000, Huang 1995), which makes no assumption about the classifier dependence.

A neural network can also achieve OCR combination. We use as input two vectors V_{alpha} and V_{error}. V_{alpha} is the confusion vector of each vocabulary character, whereas V_{error} is the error status of the character according to the error types among: substitution, deletion, addition, etc. Its component values belong to {0,1}. The network is able to perform a simultaneous analysis of the error and the label that allows a good generalization, contrary to the BKS method, which would need more samples to reach similar performance.

We have experimented these three combination methods. For each character c and each method, we obtained as result $V_m(c)=\{\tau_i\}_{i=1,N}$ where τ_i is the confusion rate for the class I, and m=1,3. For each method m, the best class C_m is equal to C_m= argmax{i, τ_i} $_{i=1,N}$, the more interesting method being m_0 defined as m_0 = argmax{m $\forall$c | c=C_m}. The final result is, for each character c, the class C_{m0}, and for a text, the list LC_{m0}.

Error extraction and categorization

OCRs commit several types of errors as: confusion, addition, deletion or segmentation. Errors are detected by a comparison between two character lists: L_{GT} representing the ground truth and LC_{m0} obtained by OCR combination. As the objective is to optimize the list alignment, we have used dynamic programming.

Let A, B, C, D characters and let # be the alignment character, the errors can be categorized as expressed in the following rules:

Confusion:	A	→	B
Addition:	#	→	B
Deletion:	A	→	#
Fusion:	AB	→	C#
Cutting:	A#	→	CD

These error types are easy to determine when the error chains are small, usually occurring in clean documents. Inversely, the errors are very difficult to locate when the erroneous length chains are large (i.e. occurring in complex documents). The error covers several contiguous characters making the error type difficult to determine, as the correspondence between characters is not obvious. The problem becomes how it is possible to locate the error origin in this long chain. (i.e. what is the rule for each character responsible of this error).

The idea is to base on small errors to detect the biggest ones. The errors are located recursively based on the erroneous chain lengths. The procedure starts by locating the small erroneous chains in the entire document. If one of the found errors occurs in the largest erroneous chain, this chain is divided in two parts: prefix before the known error, and suffix after the error, which are both recursively analyzed according to other smaller errors detected in the document. It is obvious that this approach can work only when the erroneous chain length is reasonably large.

Once the errors are detected, we generate for each of them a probability function defined by: M_c: confusion matrix, M_f: fusion matrix, M_s: cutting matrix, V_a: addition vector, V_d deletion vector.

Error correction by ICR

We use a local classifier, called ICR, for error correction, acting directly on the image pixels. This ICR is a modified multi-layer Perceptron with convolutional layers and specific connections between these layers facilitating the correction as seen later.

Concerning the correction itself, as the ICR is just an image recognizer, the only error that could be taken into account is confusion. However, as we don't know if the image corresponds to a character image or to a character portion occurred as a result of a segmentation problem, the confusion matrix M_c is weighted by the addition and deletion vectors. The other errors are not integrated because they don't intervene directly at the character level (i.e. image level).

The erroneous character image is taken into account by the ICR if and only if the maximum of the confusion rate is lower than a fixed threshold S (i.e. $\max_i (M_c(i,C_{m0})) < S$). Contrarily to the OCRs that operate as black boxes, the ICR topology and functioning are adapted according to the error type.

Concerning the ICR architecture, it is based on a convolution neural network like LeNet (Lecun, 1998). It is composed of five layers:

- The first one corresponds to the input image, normalized by its center and reduced to 29*29.

- The next two layers correspond to the information extraction, performed by convolutions. They are described as follows:
    - The second layer is composed of 10 maps; each one corresponds to a specific image transformation by convolution and sub-sampling reducing its size. For each map, all the neurons have the same input link number and share their weights.
    - The third layer is composed of 50 maps; each map represents the convolution of a combination of five maps in the previous layer. Here also, in each map, the link weights are shared by all its neurons. There are several methods to achieve the weight sharing within a map: 1) to create an external map corresponding to a receptive field: each neuron has its input linked to this field; 2) to consider a pivot neuron in the map that synthesizes the receptive field. In this case, an input link is described by Link(n,l,ξ(w)) where n is the neuron number of the previous layer l and ξ(w) is a link to the value of the weight. Usually, when there is no weight sharing, ξ(w) = w, each link has its own weight.
- The last two layers are specialized in the use of two kinds of information: information extracted by the previous layers, and information given by the confusion matrix according to OCR combination result (i.e. the matrix column). They are partially connected; only neurons specialized in the classes presented in the confusion matrix are connected. More precisely:
    - The fourth layer is composed of many neuron blocks; each one of them represents a character class. This allows us to dispose of many neurons sharing the information and consequently voting for each class.
    - The last one corresponds to the output.

Error Correction Principle

We use the semantic link between layers to operate the correction. In fact, as each ICR layer represents a transformation and as the entire layer entries are equivalent (i.e. of the same type, with the same link number), it is possible to change the transformation meaning by transforming the links. To achieve such a transformation, we replace the first ICR layer by a self-organizing map (SOM). The physical location of the SOM neurons and their values (coordinates) are identical. This means that when the SOM is transformed (i.e. neuron location is changed), each neuron of the second layer remains connected to the SOM neuron in the initial physical position, but with different values, representing the new link.
The total use of the SOM is shown in Fig.2. When a failure happens in on the input character, signaled at the MLP output, the SOM is trained on the erroneous character leading to the modification of the initial position of the neurons. As a consequence, the connection links between the first layer and the second layer are modified.

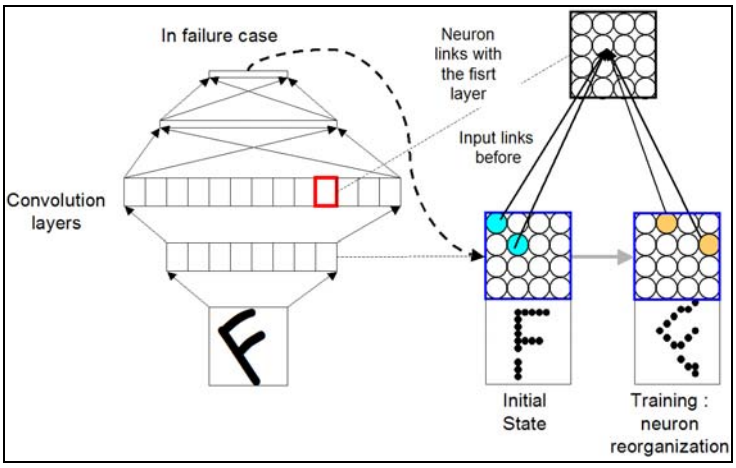

Fig.2 Global Schema of the SOM transformation

SOM training

SOM training is operated in several steps as follows:

- Step 0: SOM initialization: each neuron in the map is represented by a vector V ={x,y,v} where x,y are the real coordinates of the neuron and v is the value of the corresponding neuron in the input layer.
- Step 1: Base creation: we select all the image pixels where values are greater than a threshold, i.e. all the black pixels.
- Step 2: training: for each pixel in the image, we choose its closest neuron in the SOM according to their Euclidian distance. Then, we select the neighborhood for neuron weight updating. Instead of classical methods using a circular neighborhood around the winner, only neurons present in the sector which origin is the neuron winner and the extrema is the ideal position (corresponding to the pixel position in the image) of the neuron are updated. This is to favor the direction between the winner and the ideal position of the neuron in the SOM. The sector angle is a threshold given manually.

Experiments

The system has been tested on law documents. As many solutions are possible to create PDF file, we here use the raw image as input of our system. We show the result obtained on a law document composed of 5000 characters, at different resolution: 150dpi and 300dpi. The results first show the different OCR behavior for each resolution and the improvement of the OCR combination: the number of confused character decreased for each combination. The BKS combination offers the best reliable result. The Oracle voting simulating an optimal vote shows that both OCR can't cover by their result a perfect recognition, proving the

interest of completing the combination by the ICR. Among the characters rejected, many accentuated characters are present, like for the confusion between ò, ó, ô, ö and o. In this case, the OCR combination is completed by the ICR; its topology focuses just on the accent and not on the whole character. In another example, the confusion between "‘" and the mark "1" can perturb the logical structure as the mark is a reference to another part of the document; this mistake can damage the next reformulation step.

	Recognition	Rejection	Confusion	Addition	Deletion
(150dpi)					
OCR 1	96,63	0,02	1,21	0,48	1,37
OCR 2	71,79	0,10	6,78	1,84	16,79
Majority Voting	69,24	27,69	0,06	2,85	0,16
Oracle Voting	99,29	0,69	0,00	0,02	0,00
BKS	98,02	0,00	0,40	0,83	0,71
(300dpi)					
OCR 1	96,96	0,00	1,10	0,41	1,32
OCR 2	99,61	0,00	0,21	0,04	0,02
Majority Voting	96,86	2,61	0,04	0,48	0,02
Oracle Voting	99,71	0,25	0,00	0,04	0,00
BKS	99,88	0,00	0,06	0,06	0,00

Table 1 Recognition rate for OCR and their combination

STRUCTURE RETRO-CONVERSION

Outline

Retro-conversion task aims at finding the logical structure of the document. This structure will be very helpful for metadata extraction needed for document retrieval process. When documents are not structured, like those obtained by OCR, a structure retro-conversion step is needed to recover such structure. This operation is not straightforward from the layout as the two structures are not bijective. Researchers who tried to mechanize this step by using syntactic parsers or rule based systems like (Anigbogu 1993, Brugger 1998, Hu 1993, Bouletrau 2000) quickly realized the task hardness even for documents in Pdf form (Vincent 2001, Chao 2001, Chao 2003, Hadjar 2004, Anjo 2001). Chenevoy (Chenevoy, 1991) brought a partial answer to the problem by using a physical-logical duality expression model and an expert system using this model as a knowledge base, but remained limited to some of well-structured documents.

We propose a new approach that is not based on fixed rules suggested in advance. This approach learn how to establish the relationship between physical and logical structures and should be enough flexible to accept physical structure variations. The neuronal model can

satisfy these conditions: it is less sensitive to noise and can learn links between physical perception of primitives and logical structure determination.

We have chosen a similar neuronal model from McClelland and Rumelhart (McClelland, 1981), called Transparent Neural Network (TNN). This model was already used for word perception. It makes use of logical interpretation decomposition in levels going from the local view to the global view. It contains three interpretation levels: local level corresponding to the primitives, intermediate level that represents the letters (first stage in logical interpretation), and global level (final stage in the logical and whole interpretation of the word). This model achieves the word perception into two movements: propagation of the physical towards the logical (of the primitives towards the word) and retro-propagation of the logical towards the physical (of the word and thus of the context towards the primitives). This context return was specified better in Côté's work for handwriting recognition (Côté, 1997) by a window-slicing system that specifies the place where the primitives should be placed within the word image. Maddouri (Maddouri, 2002) proposes an additional level of syllables between letters and words. For the context return, she proposes in addition to the windowing a more suitable system of primitive extraction based on Fourier descriptors.

Model overview

We propose to use such a model for structure retro-conversion. As the logical structure is not a straightforward operation, the structure is achieved during several perceptive cycles. A perceptive cycle comprises two movements. In the first movement, the system extracts a first group of physical indices, presents them to the TNN, which, by propagation, gives an output vector with recognition rates. This vector is then analysed to decide the relevance of the solution obtained. If no pattern has a score raised enough compared to the others, the system signals an ambiguity. Then the system operates a context return to find the causes of this bad recognition. When the faulty entries are detected, the extraction algorithms are parameterised again in order to correct or refine the input. A new propagation is achieved and the new output is compared to the old one. If ambiguities decrease, one remakes one complete cycle. If ambiguities persist, the system then decides to make use of another group of physical indices to bring more information and to try to better differentiate the classes.

The group chaining and the index correction are done especially according to extraction speed criteria. Our idea is to limit the quantity of computation according to the granularity of work to be carried out. Indeed, certain zones are easier to recognize than others and it is not always necessary to launch all extraction algorithms to identify certain patterns. Sometimes, few and easy computing information (as position or bounding boxes area) is enough to discriminate a pattern. Similarly, if the sub-pattern location is known, we can focus on it during the context return to extract the only physical indices allowing us to confirm the pattern or to resolve the ambiguity between several patterns. One can thus imitate the human behaviour while launching the good tools for extraction on the good zones of observation.

The second phase, detailed in the next section, is a semi-automatic classification process. It is a step added to the system in order to take into account the TNN rejections and to treat the patterns that are not in the learning base.

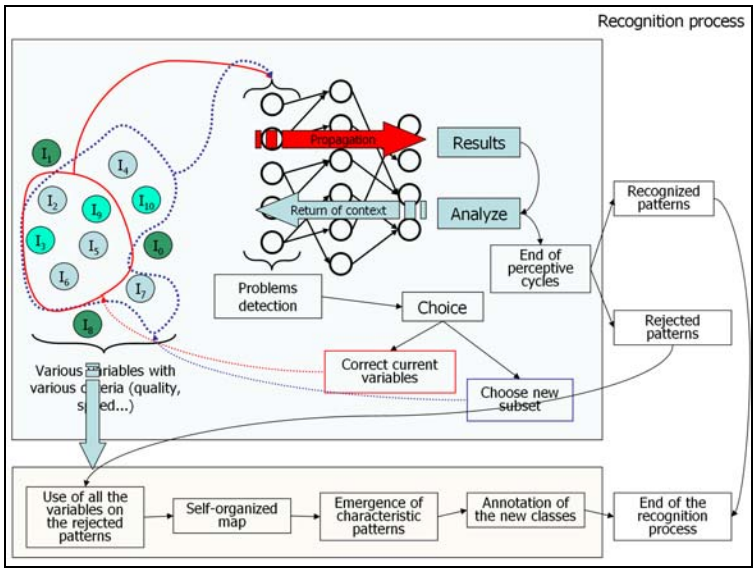

Fig.3 System overview

Improvements

We brought improvements to the recognition process. They relate to network training, architecture definition as well as to input data categorization.

- The training phase was performed manually in the previous systems by imposing connection weights. That was possible because the data is far from numerous and shared not very complex relations. Wanting to generalize the process, we operate a machine learning process but by limiting the procedure of retro-propagation to each layer because of the network transparency.
- Concerning the architecture, we are based on common concepts valid for any document and possibly on DTD when it is present. This knowledge is translated manually to correspond to the spirit of the local-global vision of the network and its context return.
- The input data are categorized in order to adapt the quantity of work to the pattern to be recognized. It constitutes the most original part of the system and will be developed in the following paragraph.

Data categorization

Data categorization corresponds to a kind of classification spreading out the data according to certain criteria, thus allowing the system to refine its recognition rate. The criteria selected correspond to the extraction speedup and the data informative power.

We use methods stemmed from data analysis area such as Karhunen-Loeve transform for space reduction, operating in two steps. First, the correlation matrix CR_{Mat} of the input vectors X of the training basis is determined. Second, CR_{Mat} is decomposed into $V*L*V^T$ with V containing the CR_{Mat} eigenvectors and L the diagonal matrix comprising the eigenvalues in descending order. The KL transform stops choosing q first columns of V to reduce the space X in the reduced space $Y=V^T*X$. The choice of the greatest eigenvalues guarantees the minimization of the prediction error of Y towards X but can choose variables containing redundant information (Balci, 2002). We should not consider the eigenvectors independently to build a group. As we want to use all the variables, it is better to build groups containing "complementary" variables i.e. that represent different axes.

In (Sun, 2004), the author uses a genetic algorithm to find the best variable set and use his classifier for solution evaluation. Contrary to Sun, we would not like just one set but rather to gather all the variables in sets. Moreover, we will avoid using, like him, the recognition system to operate such a classification: one finds initially the groups by an independent mechanism, and then we build TNN according to these groups.

The partitioning algorithm functions as follows: if one notes V_i the vectors lines of V, two variables *a* and *b* are correlated if their vectors V_a and V_b are close by the Euclidean distance. One uses K-means or a Kohonen map of k neurons N_i to classify the V_i in k classes. Once the classification is finished, one seeks to create the groups of variables x_i. These groups are formed in a progressive way according to the distance from V_i to the gravity centres of their classes: the 1st group contains the elements closest to each class. There are thus k groups, each one have complementary variables. Then, the groups are sorted by descending order of predominance of their variables (see Fig.4).

Once these groups are formed, one can either use them just as they are to form the groups to be used in the perceptive cycles because in general the most informative variables are the most difficult to extract or, as in our case, to consider that the extraction time of each index is not constant. We can then either arrange the groups according to the total extraction time of each one, or to remake other groups of variables according to time of extraction and by regarding the membership of a variable to an old group as indicator of predictive capacity.

When the final groups are formed, one creates as many TNN as groups (3 in our case). The first network uses the first group, the second TNN uses the groups n° 1 and n° 2 until the last which uses the whole of all the indices. We use the training technique mentioned in the

preceding section on each network in order to create different classifiers that simulate different vision levels.

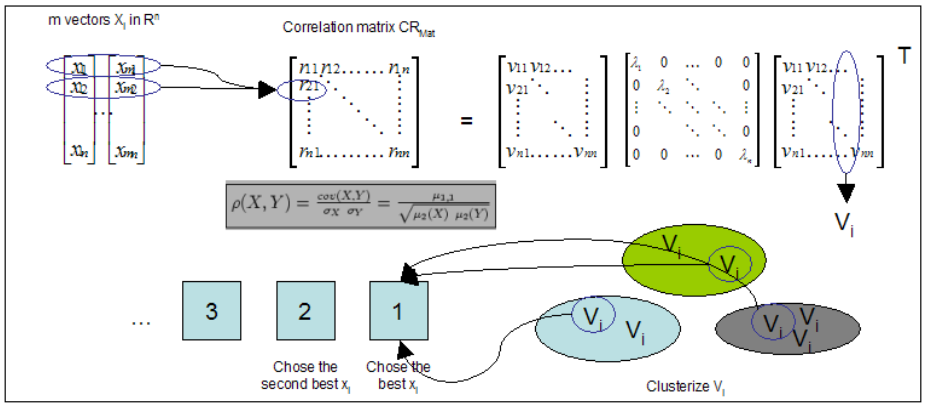

Fig.4 Categorization schema

ANNOTATION AND INDEXING

Once the physical and logical structures are determined, it is necessary to build the third level of an electronic document as defined by V Christophides (Christophides, 1998), named "semantic" level. It is a question of locating and of highlighting for a future use certain information of the semantic type. This information relates at the same time to the whole document (metadata) and to its contents (semantic tags). This step can be carried out by the expert or automatically. This information is of two types:

- Metadata which classify the document, describe it from the point of view of its field and its topics resulting from thesaurus.
- Semantic tags which delimit semantic information and which one can oppose to the logical tags which give a logical.

Being given that we wish to add to documents semantic tags during annotation, we have needs for two types of information on the tags that a DTD does not contain:

- The tag type:
  - a tag is logical when it is used to describe the logical structure of a document, i.e. the document organization and reading. Ex: <title> It is the title </title>
  - One will speak about semantic tag when it gives a particular sense to its contents. Ex: <specy> Mammal </specy>
  - Lastly, a typographical tag gives information on the content formatting. Ex: <bold>Written in bolds</bold>. A combination of these types is completely possible.

- Relationships between tags: all the tags of tagged document have at least one relationship between them. It is about the implicit relation father-son connected to the tree structure. We define other types of relationships:
    - *generalisation / specialisation*: it is about the link linking two semantic tags of which one is under-concept or father-concept of the other. This relationship concerns the tag direction.
    - *composition / part of:* is the relationship between a semantic tag which is made up of or started from another tag from the sense point of view.
    - *Semantic dependence:* is the relationship between a semantic tag whose content can only be understood in relation with another tag.

FILE CONSTITUTION

Document Selection and classification

Introduction

In response to a user request, the document retrieval engines turn over sets built more or less well and ordered according to criteria of relevance. The experiment showed that neither the relevance nor the linearity of presentation are factors sufficient for the user because 1) they do not make it possible to have a global and synthetic vision results, 2) certain documents can escape the criteria from relevance either due to under-referencing (innovation) or due to inadequacy between the terms of the request and the indices which describe them. It thus appears necessary for the task of file constitution which one wishes to carry out, of going at the engine frontiers to have a document set much more informative. The idea is to accompany these engines by tools of filtering and sorting of their results. The solution suggested in the literature for the accompaniment is 1) to operate a theme classification which allows an organizational vision of the results (Hearst, 1996) and 2) to exploit the interest related by the users to the documents to carry out a filtering much more relevant correcting the referencing problems.

To obtain a theme classification set, the literature proposes two possible approaches:

- **Categorization** which consists, starting from a preliminary manual indexing (realized by an expert of the field) attributing thematic to documents, to reorganize the documents in topic class. This categorization is left to the specialist responsibility and was not the subject of a particular formalization (absence of generic method).
- C**lassification** which goal is to discover the groups (clusters) of similar documents, to make emerge "latent" classes of a document set. Not supervised classification is often used because one makes classification with stolen not allowing the intervention of the user. There are several not supervised methods of classification:
    - Methods based on similarity calculation between documents, like the dynamic clustering techniques: K-means (Rocchio, 1966), or hierarchical grouping

methods like: HAC (Hierarchical Agglomerative Clustering) (Voorhees, 1986), Suffix Tree Clustering (Zamir, 98), Semantic Online Hierarchical Clustering (Zhang, 2001);
- Probabilistic methods;
- Neuronal methods like the Self-organizing Maps.

However, this type of classification per categorization or not supervised does not allow to integrate the user experiment, which is insufficient for the constitution of personalized files. One thus needs an approach more centered around the user who more takes into account his work practices and his document vision. It is necessary thus that the user can create his own categorization.

To measure the interest carried to the documents by the users, the literature proposes to use the concept of "collaborative filtering". The goal is to emphasize the majority user opinions on the documents which will be then used like personal recommendations. The system listens to each consultation, characterizes it and proceeds to a notation which reflects the general opinion.

Proposed Solution

We have chosen to combine automatic classification and categorization approaches (by the specialist and the user). The process starts from the information retrieval result, with a categorization which let emerge the key topics in connection with those given by the expert during the indexing. These key topics or index being characteristic of the documents without being inevitably in connection with the request; we then continue the categorization by procedures of not supervised classification which will seek other key topics more stressing the adequacy between the document (summarized) and the request (key words). The advantages of this approach are: 1) emergence of the current topics (very present in the classifiers), 2) minimization of the errors of each one of these techniques, 3) increasingly significant contributions of the user news (knowledge on the document). At the end of the process of classification and for each topic, the documents results in the topic are classified according to the interest (note or evaluation) that the users expressed to them.

Description of the search result

From terms extracted from the request, documentation base (B) is questioned by using a traditional search engine. The result is a set of documents noted R where the d_m are quadruplets describing a document result (identifier, addresses, title and summary built "on the fly", in adequacy with research).

Document categorization

We exploit metadata suggested, for example, by Dublin Core for a standardized description of the leading structures of the documents, MARC for the bibliographical structures, or more precisely in our application, XML citations describing the EEC papers, referring more to the legal topics of this specific base.

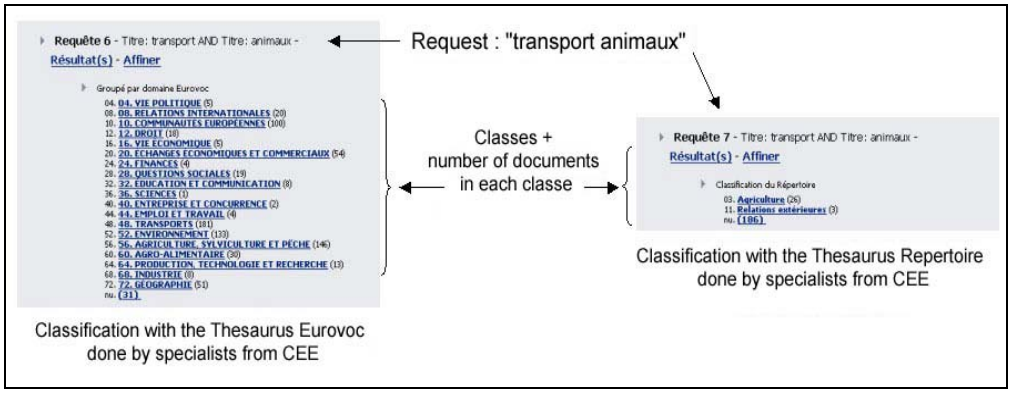

Fig.5 Expert Categorization

Non supervised classification

For classification, currently, we call upon three algorithms, based on following solutions:

- Hierarchical Agglomerative Clustering (Voorhees, 86). In this approach, each document in the collection or list is treated as a cluster and added into a pool of available clusters. Consequently, with *n* documents, there are initially *n* clusters. Next, all pairs of clusters in the pool are compared and the most similar pair is selected. Document/cluster similarity is typically computed using a metric such as the Cosine, Dice, or Jaccard formula (Rorvig, 1998). The most similar cluster pair is then merged into a single cluster, and added back into the pool of clusters. The HAC algorithm implemented used the similarity metric of word intersection. Word-intersection clustering (Word-IC) was defined by Zamir, et. al. in the context of snippets from web documents (Zamir, 1997).
- Suffix Tree Clustering (Zamir, 98). The algorithm is interested in sentences common to the documents. After pruning of blank words, it determines the roots of the words (stemming), identifies sentences of each document, then creates a reversed index of the sentences by using a suffix tree. The same document can belong to several classes. The sentences are balanced: the score of a sentence depends on the number of words which it contains as well as number of documents in which it appears. Each sentence constitutes a basic cluster. The following stage thus consists in amalgamating these basic clusters according to a function of similarity.

- Lingo (Osinski, 03). The method consists in discovering the key labels initially, then to assign the documents in each one of these groups. The key labels answer a certain number of principles like: to appear in a minimum number of documents, not overlapping between them, to be a complete sentence, etc.

These algorithms were applied on the basis of documents without particular adaptation. The result is a set of descriptions and documents associated. Each algorithm enables us to obtain a whole of classes containing the keyword and the documents which are attached to it. Fig.6 shows the result of the not supervised classification obtained by algorithms LINGO and HAC starting from the request: "animals transport".

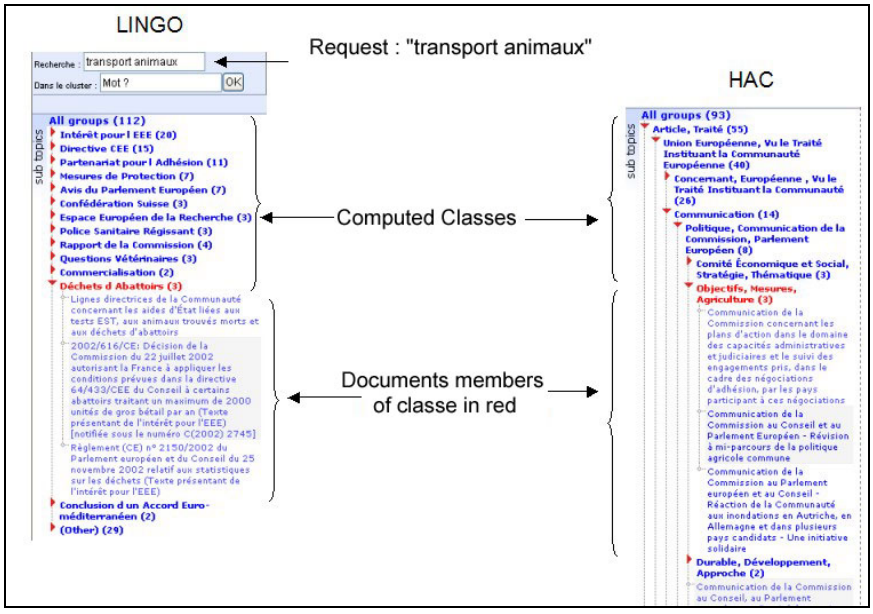

Fig.6 Results of non-supervised classification

User Classification

During result consultation, the user is invited to give an appreciation on the documents (knowledge on the document) by associating to each document a set of keywords more in connection with his own vision of things (practices of work, interest for its research, etc). This information is stored in a database and classified by user like by the same user group profile. This enrichment action is done by the user, either during the consultation or at the file closing, by giving an instantaneous appreciation in the form of note or topic chosen freely. The information search enables us to enrich the theme list sets with the information brought by the user, his group or the whole of the users.

Classifier Combination

At this level, one has a classification set $\zeta_{i,\ i=1,n}$ where $\zeta_i=\{C_i\}$ where $C_i=\{m_i,E_i\}$. m is a description or a keyword and $E=\{D_k\}$, the document set of this topic, according to the application. The ζ_i can come from metadata, automatic classifications or from classifications carried out by the user. For the ζ_i stemmed from metadata, within the framework of the Eurlex base, there are 3 possible repertories (ζ_M): 1) the Eurovoc keywords (multilingual thesaurus covering the fields in which the European Communities are active), 2) the repertory classification (another type of table of index), 3) table of contents.

From some m_i of all the C_i classes, in all classifications ζ_i, one extracts the keyword list $L=\{m_i\}$ and one creates the matrix of the word vectors following the vector model initiated by (Salton, 1971)

$$D_{i,j} = \sum_{k=1}^{n} (D_i \in \zeta_k(m_i)) \times p_k$$

p_k is the weighting given to the classification algorithm. It is initialised at the beginning by the expert, it can be then modified by the user according to the result appreciation. The factor $(D_i \in \zeta_k(m_i))$ is a boolean equal to 0 or 1 according to whether the D_i document belongs or not to the class C of the classification ζ_k and which contains the keyword m_i.

Lastly, we use the Kppv method to define the clusters.

Document notation

Once documents are classified, we seek to allot to them a final note (global) of interest dependent on several elementary notes, allowing to order them in their classes. Let L be the document list of a class C: $L = (d_1 \ldots d_n)$ where $d_i \in C$.

Final note N_F depends on three elementary notes:

- note allotted by user (N_U);
- average note allotted by the user group (N_G);
- average note allotted by the set of users (N_A).

The user note N_U translates the interest which this on carries to the document and is deduced from a set of actions carried out by him on the document: the opening hour of the page, moment of closing, the addition in its mark page, bonds visited starting from this page. One uses the function of Chan (Chan, 1999) to calculate this value (Interest):

$$Interest(Page) = Frequency(Page) \times (1 + IsBookmark(Page) + Duration(Page) + Recency(Page) + LinkVisitPercent(Page))$$

The user note U on document i will be called n_i. N_U is the user note vector $N_U = (n_1...,n_n)$ where $n_i = 0$ if the user did not consult the document.

The group note N_G is the user notes average of the group. A user is taken into account only if he consulted the document. Let G be which represents all the user vectors notes the same group and N_G the vector of the document average notes of the group:

$$G=(N_{g1},\ldots N_{gm}) \text{ where } N_{gi}=(n_{i1},\ldots,n_{in})$$

The average group is calculated as follows:

$$N_G = (m_1,\ldots,m_n) \text{ where } m_i = \sum_{k=1}^{m} n_{ki} \;/\; \sum_{k=1}^{m} (n_{ki} \neq 0)$$

In the same way, one builds N_A the average notes allotted by the user set.

Final note N_F allotted to a document is deduced from the user strategy that we privileged in this work, namely: impact of the user initially, impact of its group if he is not sufficiently qualified in the field, impact of the user set if the group is not enough qualified. This results in a different weighting of notes from these sets.

The final note will thus be calculated as follows: $N(d_i) = N_U(i)*p_U + N_G(i)*p_G + N_A(i)*p_A$

Consequently, documents presented at the head of the list are those which will have interested more the user. Then come the documents whose user was not informed yet, but that the members of his group or of the user set have appreciated.

File creation

Introduction

This step functions starting from the documents selected by research and classified like sight previously. Contrary to the classification procedure which works in a global way on the documents, this step works on the level of documentary units or fragments (paragraph, article...) characterized by the DTD. One also speaks about brick of information. The file creation module comprises a set of elementary strategies allowing to select among the most relevant fragments for a user, to order them, to assemble them, to add hyper links to it then to format the whole in order to obtain a document answering the user expectations. For all the following processing, the user can intervene directly on the development or let act the system in a mode by defect (A profile is then used). We can decompose this module into three sub-modules: extraction, composition and formatting.

Fragment extraction

As we mentioned previously, the first stage of the fragment extraction consists in splitting up the documents. It is a question of cutting out these documents in logical fragments. Before describing this fragmentation, we introduce some notations:

- the sub-base $D = \{d\varphi_{i\ (i=1...n)}\}$ where $d\varphi_i$ are documents XML (physical) possibly pre-selected by a search. Each document $d\varphi_i$ is a couple (d_i, c_i) where:
  - d_i = (id, url, title, abstract) and
  - c_i is the document content, i.e. a fragment constituted by all the document fragments. It is question in fact of the root of the XML structure tree.
- $F = \{f_{j\ (j=1..m)}\}$ where f_j represent fragments stemmed from all the documents. Each fragment is a quadruplet: $f_j = (d_j, f_f, c_j, sem)$ where
  - d_j is the document descriptor (defined previously) from which comes the fragment,
  - f_f is the father fragment containing the current fragment,
  - c_j is the fragment content i.e. a fragment composed of its sons
  - *sem*: F –> F is a function which extracts from c_i the semantic sub-fragments contained in the logical fragment.

The fragmentation function is defined by chunk: D –> P (F) which to a $d\varphi_i$ makes correspond a partition $\{f_1, f_2,..., f_m\}$ of F (set of the fragments of $d\varphi_i$); The fragment base is obtained by $F = U_{d\varphi \in D}$ (*chunk*($d\varphi$)).

We give then the possibility to a user to make a search on these fragments; hence, we define a filtering of the fragment base. Let F be the $filt_{user}$: F –> {0,1} which, to a fragment, makes correspond 0 if this fragment is interesting for the user, 1 if not. This function is that of the base filtering. The set of n selected fragments constitute a n-uple $(f_{i\ (i=1..n)})$ where $filt_{user}(f_i)=1$ $\forall i$. The singleton $F_{user} = \{(f_1, f_2,\ldots,f_n)\} \subset F^n$.

Research is carried out by keywords or by the value of the semantic contents (identified at the time of the annotation) of the logical fragments. These logical fragments come from F and with constraints on the granularity and/or the topics which interest the user.

The granularity is the name of the logical tag delimiting the interesting fragment (ex: “article” in the EC Official Journals). This one is in the profile or given by the user. The topics are chosen by the user in a list proposed during the fragment extraction. They correspond to the topics synthesized during the research/classification process presented previously.

The semantic content values depend on the logical fragment extracted (granularity) and from its under semantic fragments; the function *sem* of the fragment allows to find these contents.

According to types' of the semantic fragment values, research interfaces will be proposed to a user. For example, for a semantic fragment <date>, a search interface for logical fragments per date will be proposed. We identified 3 types of semantic values: dates, numerical, literals which induce three types of possible researches.

The advanced research type offered depends consequently on the DTD. Research adapts to the user desires.

The function index $filt_{user}$ reminds that this filtering function uses the profile and the parameters preset by the user to filter the fragment base F. We use this index each time that a function is parameterised by the profiles and/or information available via the user interface.

Fragment recombining

The fragment recombining is the constitution starting from selected fragments of a new fragment which contains them all and which will be the content of the document result. This new logical fragment is obtained by ordering the fragments which make it up according to various criteria, by adding logical fragments such as titles, index, links. Let:

- rec_{user}: F^n –> F the fragment recombining which transforms a fragment n-uple into a fragment which contains the whole of the ordered fragments, with new elements of composition (new fragments: titles, links, indexes...). Parameter n depends on each document. This function can be seen like a composition in the functional sense: $rec_{user} = ass_{user} \, o \, ord_{user}$ where
  - ord_{user}: F^n –> F^n is related to scheduling which with a tuple (f1, f2..., fn) made correspond a n-uple $(f_1, f_2, \ldots, f_n)$ with $i_{k\,(k=1\ldots n)} \in \{1, 2, \ldots, n\}$ and $i_k \neq i_{k'}$ for $k \neq k'$
  - $ass_{user:}$ F^n –> F creates a new logical fragment starting from an n-uple while adding to it composition elements. These composition elements are logical fragments created starting from existing fragments (links) or not (titles), they can be inserted everywhere in the document.

A virtual document can be now defined by $dv = (d_i, c_i)$ where

- d_i = (id, url, title, abstract) and
- $c_i = rec(f_1, f_2, \ldots, f_n)$ is the document content

The functions described above take account of the user desires and expectations. These desires are expressed through an interface and/or a profile.

Several criteria intervene in scheduling. It is possible to sort the fragments according to the value of the possible semantic elements or by the topic importance. For each of the 3 types of semantic values, we define an operation of sorting: by dates, by numerical values or by lexicographic sequence. Each sorting can be bottom-up or top-down and has a priority

(useful for the sorting combination). Another possibility of scheduling uses the topics generated at the time of the search/classification and the note allotted by the user to classify the logical fragments coming from these topics.

New fragments addition consists of link addition between keyword occurrences on the documents sources of the fragments or others; titles which are actually the topics synthesized at the time of research classification. Through the interface, the user can directly publish the fragments without being aware of these fragments.

The fragment extraction and recombining thus allow the creation of one or more virtual documents possibly dependent between them or related to the documents sources. The whole of the documents created and those obtained at the time of research classification forms a file: $Dv = \{dv_{i\ (i=1..m)} \mid dv_i$ are possibly dependent or linked to their $d\varphi_j$ source by hyperlinks$\}$

Document formatting

After recombining, the user posts his file on the screen and possibly safeguards it. Posting takes account of the user expectations and this one has the possibility through the interface to modify the font style attributes (typographical elements like size, colour, ...) and their range (all, part of the document) and layout (one or more pages, column number if the selected output format is HTML).

The safeguard transforms $Dv = \{dv_{i\ (i=1..m)} \mid$ the dv_i are possibly dependent or depend on others by hyperlinks$\}$ in $D\varphi = \{d\varphi_{i\ (i=1..m)}\}$ where the $d\varphi_i$ are files of a certain format chosen by the user (thanks to an interface or to its profile). The user can choose among several output formats: XML, HTML, PDF, RTF, TXT. The selected output format as all the formatting parameters (police forces, styles...) determine the generation of an XSLT stylesheet which will allow the posting of document XML in this format. This posting will be dealt by external software, such as a navigator.

For each possible format, an XSLT stylesheet is written by the expert or by all the DTD tags; this stylesheet for each output format is indirectly related to the initial profile. The personalization of the stylesheet is carried out when the user made a safeguard with the format defined in the profile or that it chose. The preset stylesheet is then recopied and possibly modified by preserving only the links present in the virtual document to format and by using the formatting information defined in the profile or specified by the user via the interface.

$style_{user}$: $Dv \rightarrow D\varphi$ is the personalization function of the stylesheet. It takes into account the user profile and/or the parameters determined by him through an interface and links contained in the virtual document. Only those are the subject of rules of the XSLT sheet. Thus, it is possible to change style for each document of the file.

By generating the stylesheet, the system introduces if necessary the formatting objects (FO). When the desired output format is HTML, a processor XSLT formats the document according to its stylesheet and produces a HTML document which can be posted thereafter in a Web navigator. If the output format is PDF, the processor XSLT generate a new document XML, maintaining type XSL-FO and an external processor XSL-FO creates a final document PDF. In practice, these three sub-modules (extraction, composition, formatting) are carried out in the two parts below:

- Tools part: it is about a toolbox of fragment extraction tasks according to several criteria, of fragment handling, formatting. These tasks are republication tasks; they are the main functions described previously. They are parameterised thanks to the interface profiles
- Intelligence part: according to a user need expressed through the interface or the profile (or both), it generates a combination of republication tasks. It is here about an expert system since one can suppose that the creation of a document follows precise rules depending on the field although these rules are difficult to identify clearly. These rules allow the expert system to control the session functioning and to carry out certain necessary pre-tasks if the user wishes to carry out an operation without to have carried out the intermediate tasks, to combine the information entered by interfaces with those of the profile.

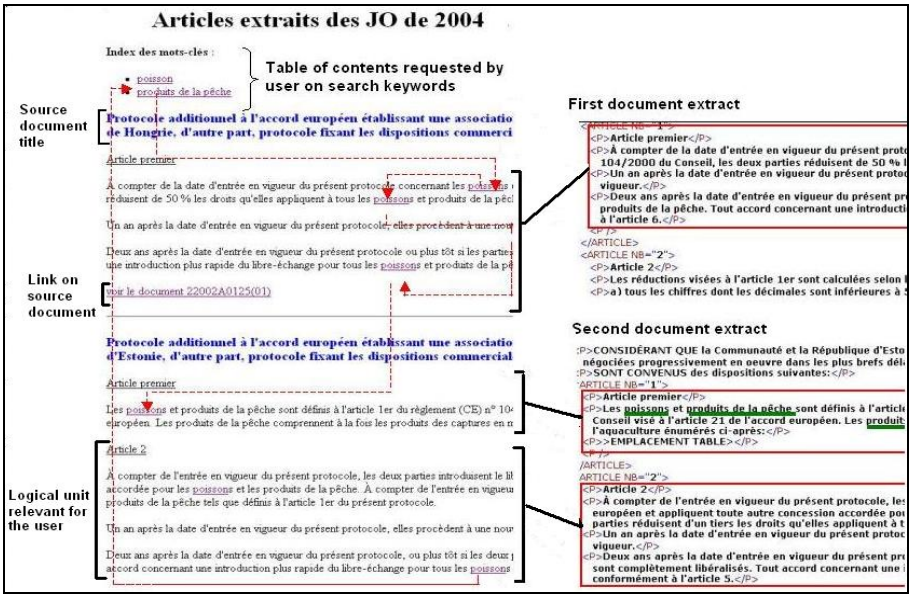

Fig.7 A simple reedition result based on European Union documents. On the left, the resulting html document and on the right, the source Xml documents used for composition

CONCLUSION AND FUTURE WORK

We believe that with current project PAPLOO we may help general users of big sets of documents to process and build personalized repositories of documents. PAPLOO will manage general constraints on these sets of documents, and will provide all kinds of users with easy tools to integrate seamlessly non-homogeneous sets of documents and related information, as they may be extracted from central and organizational databases. PAPLOO is the result of a tight partnership between research, university laboratories specialized in documents recognition, management, and a firm specialized in Enterprise Content Management (ECM). Some tools are nowadays at our disposal. The next step is to integrate these tools on a central platform containing many millions of documents, corresponding metadata and categories, and to simulate different kinds of usages.

REFERENCES

[1]. Christine M. and Lainé-Cruzel S. (1999), "Profil-Doc : Un prototype de système de recherche d'information personnalisé selon le profil des utilisateurs. Atelier sur les Documents Virtuels Personnalisables : De la Définition à l'Utilisation". *11ème Conférence Francophone sur l'Interaction Homme-Machine, IHM'99*.

[2]. Iksal S. (2002), "Spécification Déclarative et Composition Sémantique des Documents Virtuels Personnalisables". *Thèse de doctorat*, Ecole des Hautes Etudes en Sciences Sociales, Brest, France.

[3]. Watters C. and Shepherd M. (1999), "Research issues for virtual documents. Dans Proceedings of the Workshop on Virtual Documents, Hypertext Functionality and the Web". Toronto, Canada. Eighth International World Wide Web Conference.

[4]. Brusilovsky P. (1996), "Methods and techniques of adaptive hypermedia". User Modeling and User Adapted Interaction, Vol. 6(2-3) : 87-129.

[5]. Ranwez S. (2000), "Composition Automatique de Documents Hypermédias Adaptatifs à partir d'Ontologies et de Requêtes Intentionnelles de l'Utilisateur", Thèse de doctorat, Université de Montpellier II, Montpellier, France.

[6]. ICWE (2004), http://www.icwe2004.org/

[7]. Bahler D and Navarro L (2000), "Methods for Combining Heterogeneous Sets of Classifiers", 17th Natl. Conf. on Artificial Intelligence (AAAI 2000), Workshop on New Research Problems for Machine Learning.

[8]. Gunes V, Ménard M, Loonis P and Petit-Renaud S (2004), "Systems of classifiers: state of the art and trends", *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI), 17(8), World-Scientific.*

[9]. Suen C.Y and Lam L (2000), "Multiple classifier combination methodologies for different output levels", Springer-Verlag Pub., Lectures Notes in Computer Science, Vol. 1857 (J.Kittler and F.Roli Eds.) p. 52-66.

[10]. Huang Y.S, and Suen C.Y (1995), "A method of combining multiple experts for the recognition of unconstrained handwritten numerals", IEEE Trans. On Pattern Analysis and Machine Intelligence 17(1), p.90-94

[11]. LeCun Y, Bottou L, Bengio Y and Haffner P (1998), "Gradient-Based Learning Applied to Document Recognition", *Proceedings of the IEEE, vol. 86, n°11*

[12]. Anigbogu J. C., Belaïd A. and Chenevoy Y. (1993), "Qualitative analysis of low-level logical structures". *Electronic Publishing*.

[13]. Brugger R., Bapst F. and Ingold R. (1998), "A DTD Extension for Document Structure Recognition". *Electronic Publishing*.

[14]. Hu T. and Ingold R. (1993), "A mixed approach toward efficient logical structure recognition from document images". *Electronic Publishing*.

[15]. Boulétreau V. and Ducasse J.-P. (2000), "Automate de Reconstruction de la Structure Logique d'un Document. Application à l'Édition Électronique des Thèses Universitaires". *CIDE*.

[16]. Vincent N., Crucianu M., El Ayadi R., Faure C., Giba M. and Venet M. (2001), "From Exchange Format to High Level Document Description". *DLIA*.

[17]. Chao H., Beretta G. and Sang H. (2001), "PDF Document Layout Study with Page Elements and Bounding Boxes". *DLIA*.

[18]. Chao H. (2003), "Background Pattern Recognition in Multi-Page PDF Document". *DLIA*.

[19]. Hadjar K., Rigamonti M., Lalanne D. and Ingold R. (2004), "XED: a New Tool for Extracting Hidden Structures From Electronic Documents". *DIAL*.

[20]. Anjo A. (2001), "AIDAS: Incremental Logical Structure Discovery in PDF Documents". *ICDAR*.

[21]. Chenevoy Y. and Belaïd A. (1991), "Hypothesis Management for Structured Document Recognition". *ICDAR*.

[22]. McClelland J.L. and Rumelhart D.E. (1981), "An Interactive Activation Model of Context Effects in Letter Perception". *Psychological Review*, 88:375-407.

[23]. Côté M. (1997), "Utilisation d'un Modèle d'Accès Lexical et de Concepts Perceptifs pour la Reconnaissance d'Images de Mots Cursifs. Thèse de Doctorat". *École Nationale Supérieure des Télécommunications*.

[24]. Maddouri S., Amiri H., Belaïd A. and Choisy C. (2002), "Combination of Local and Global Vision Modelling for Arabic Handwritten Words Recognition". *IWFHR*.

[25]. Balci K. and Atalay V. (2002), "Pca for gender estimation: which eigenvectors contribute?". *ICPR*.

[26]. Sun Z., Bebis G. and Miller R. (2004), "Object detection using feature subset selection". *ICPR*.

[27]. Christophides V. (1998), "Electronic Document Management Systems", *http://www.ics.forth.gr/~christop/,* UCH/FORTH.

[28]. Hearst M.A. and Pedersen J.O. (1996), "Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results", *Actes of ACM/SIGIR Conference on Research and Development in Information Retrieval*, Zurich, Suisse, 76-84.

[29]. Rocchio J.J. (1966), "Document retrieval systems – optimization and evaluation", *Ph.D. Thesis*, Harvard University.

[30]. Voorhees E.M. (1986), "Implementing agglomerative hierarchical clustering algorithms for use in document retrieval", *Information Processing and Management*, vol. 22, 465-476.

[31]. Zamir O. and Etzioni O. (1998), "Web document clustering: a feasibility demonstration", Proceedings of the 19th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98), 46-54.

[32]. Zhang D. and Dong Y. (2001), "Semantic, Hierarchical, Online Clustering of Web Search Results.", *Accepted by 3rd International Workshop on Web information and data management*, Atlanta, Georgia.

[33]. Rorvig, M. (1998), "Images of Similarity: A visual exploration of optimal similaritymetrics and scaling properties of TREC topic-document sets", *Journal of the American Society for Information Science*.

[34]. Zamir O., Etzioni O, Madani O. and Karp R.M. (1997), "Fast and Intuitive Clustering of Web Documents", *KDD'97*.

[35]. Osinski S. (2003), "An Algoritm for Clustering of Web Search Results", *Master thesis*, Poznan Universitv of technology.

[36]. Salton G. (1971), "The SMART retrieval system; experiments in automatic document processing", *Prentice-Hall, Englewood Cliffs,* NJ.

[37]. Chan P.K. (1999), "A non-invasive learning approach to building user profiles". *Web Usage Analysis and User Profiling*.