DocMining: A Cooperative Platform for Heterogeneous Document Interpretation According to User-Defined Scenarios

Eric Clavier, Gérald Masini, Mathieu Delalandre, Maurizio Rigamonti, Karl Tombre, Joël Gardes

DocMining: A Cooperative Platform for Heterogeneous Document Interpretation According to User-Defined Scenarios

Eric Clavier[1], Gérald Masini[2], Mathieu Delalandre[3],
Maurizio Rigamonti[4], Karl Tombre[2], Joël Gardes[1]

[1]*France Télécom R&D, 2 Avenue Pierre Marzin, 22307 Lannion Cedex, France*
E-mail: rntl.ball001@rd.francetelecom.com, Joel.Gardes@rd.francetelecom.com

[2]*LORIA, B.P. 239, 54506 Vandoeuvre-lès-Nancy Cedex, France*
E-mail: Gerald.Masini@loria.fr, Karl.Tombre@loria.fr

[3]*PSI Laboratory, Université de Rouen, 76821 Mont Saint Aignan, France*
E-mail: Mathieu.Delalandre@univ-rouen.fr

[4]*DIUF, Université de Fribourg, Chemin du musée, 31700 Fribourg, Switzerland*
E-mail: Maurizio.Rigamonti@unifr.ch

Abstract

This paper describes the DocMining platform, that is aimed at providing a general framework for document interpretation. The platform integrates processings that come from different sources and that communicate through the document. A task to be performed is represented by a scenario that describes the processings to be run, and each processing is associated with a contract that describes the parameters, data and results of the processing as well as the way it has to be run. A controller interprets the scenario and triggers each required processing at its turn. The architecture of the platform is flexible enough to allow users to create their own objects, integrate their own processings into the platform, design their own interfaces and define their own scenarios. All data (documents, scenarios, contracts, etc.) are represented in XML, to facilitate data manipulation and communication inside the platform.

After a brief state of the art in section 1, section 2 presents an overview of the DocMining platform and section 3 its implementation. The next sections deal with an example of a complete scenario, for the extraction of graphical and textual parts from a gray level image: Section 4 explains how the scenario is run and shows the results it provides, section 5 describes its construction.

Keywords: Document analysis, Document interpretation, Integration framework, User interface, Scenario, XML

1 Introduction

The design of document analysis systems implies the assembly of various components and algorithms: Document image processings, physical segmentation, feature extraction, character and symbol recognition, structure analysis, interpretation tasks... This is especially true when the system has to deal not only with textual components but also with graphics and images.

In such a context, the flexible integration of all these components becomes a crucial problem. If the aim is the design of a versatile system, the components must be able to cooperate in a flexible way and the domain knowledge must be easy to integrate into the system. In the case of textual documents, the classical procedure consists in segmenting a given document into homogeneous zones and then applying character recognition methods followed by "linguistic" post-processing steps (dictionary lookup, application of language models, etc.). There is good know-how in building such systems [3], as well as in building systems applied to more general "business documents". For example, the smartFIX system offers versatility by including domain knowledge for various kinds of business documents [8].

There are also mature systems for specific domains, such as check processing [13], recognition of tables [24], or recognition of forms [18]. However, in other areas, and especially when graphics-rich documents are concerned, most of the working systems that have been designed up to now are specific to a given application area. For example:

- The system by IBM Italy for the Italian land register maps encapsulates all the domain knowledge in the core of the system [4].
- Arias *et al.* describe a system for the analysis of manhole drawings for a telephone company [2].
- Dosch *et al.* propose an interactive system for architectural drawings analysis [10].
- Several systems are designed to analyze drawings made of symbols and connecting lines [15] [19] [26].
- The MAGELLAN system exploits the connections of legends to extract information from maps [23].

Once again, all these systems are very domain specific. Using them in other domains would require considerable additional work to integrate and adapt the existing tools and components, design new ones, describe the new domain knowledge, and build up a new architecture. Only very little work has been done to build generic tools. In most cases, this implies that the system only offers low-level tools, or that it supplies complex and exhaustive knowledge representation schemes. Let us cite the work by Pasternak [21], who proposes a hierarchical and structural description coupled with triggering mechanisms for interpretation, and the DMOS method based on syntactical methods [5].

In this paper, we present an approach of the problem that may seem less ambitious, as we do not aim at representing all the domain knowledge, but that is probably more realistic when the aim is to be able to process a large number of heterogeneous documents, allowing users to define their own knowledge. This approach is based on strategic knowledge acquisition and instrumentation. The knowledge describes the image processing tools used to construct the objects and the chaining relations between these tools [22]. Our system thus allows the construction and execution of scenarios (*i.e.* sequences of processings) using processing libraries.

More precisely, the paper describes the DocMining platform that aims at providing a general framework for document interpretation[1]. The project is supported by the DocMining consortium, including four academic partners, PSI Lab (Rouen, France), Project Qgar (LORIA, Nancy, France), L3I Lab (La Rochelle, France), DIUF Lab (Fribourg, Switzerland), and one industrial partner, GRI Lab from France Télécom R&D (Lannion, France). The platform is characterized by three major aspects:

- *A document-centered approach*: Processings communicate through the document, that contains not only graphical objects and data but also traces of the execution of the processings. Such an approach avoids the problems of data scattering usually met in classical document processing chains. Moreover, developers can define their own objects and use a document structure that fits their needs.
- *A plug-in oriented architecture*: As users can conveniently add new processings, the platform is easily upgradable. Document visualization and manipulation tools are also designed according to this approach, so that a user is able to fully customize the interactions with the document structure.
- *Scenario-based operations*: Running a scenario collects the user's experience, that becomes part of the scenario itself. The scenario may then be transformed into a new processing corresponding to a higher-level granularity.

As a user can create his own objects, integrate his own processings into the platform, design his own interfaces and define his own scenarios, all (the processings of the) users share and exchange knowledge through the platform. In this way, the platform may be used for various interesting purposes:

[1]This work is partially funded by the French Ministry of Research, under the auspices of RNTL (*Réseau National des Technologies Logicielles*).

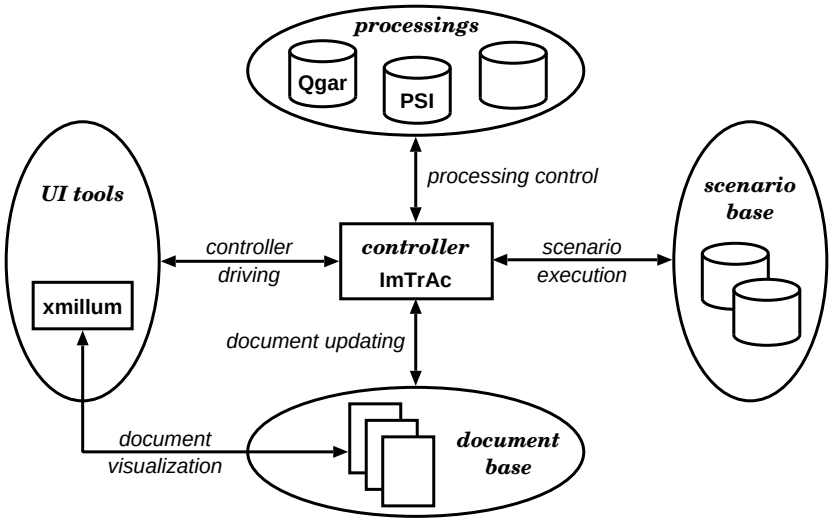


Figure 1: The DocMining platform architecture.

- Domain specific platform design: The flexibility of the architecture offers the possibility to build a platform dedicated to a particular kind of document, in which existing processings are integrated as plug-in's. The user domain knowledge is taken into account through domain specific scenarios and may be manipulated by proper interfaces, specifically defined by the users.
- Benchmarking new processings, using scenarios: A benchmarking task consists in a predefined scenario describing a full processing chain, from document acquisition to performance evaluation. Each process to be experimented is associated with a so-called contract that is provided to the users, and each corresponding algorithm is implemented according to the specifications of its contract. When the scenario is run, the efficiency of the processings can be directly measured.
- Evaluation of the quality of processing results and of the influence of the values given to parameters and thresholds, thanks to the trace recorded during the running of the processing.
- Evaluation of the efficiency of a processing according to the nature of the input documents.
- Experimentation of various implementations of a same algorithm: As users share the same integration framework, they can exchange processings. They do not have to re-invent and re-implement existing processings, already implemented by other partners.

2 Overview of the DocMining Platform

By underlining the notions of document, processing and scenario, the DocMining platform tackles the problems of the reuse of data, of processing and of strategic knowledge, which are major components of an interpretation system. The platform architecture is especially designed to take these three aspects into account. It proposes various solutions to manage the capitalization and reuse of knowledge.

Figure 1 gives an overview of the platform architecture. It includes four main entities, that are designed in terms of extensibility and reusability:

- *A document base*: Providing tools to easily access and manipulate document bases is one of the goals of the DocMining project. In our approach, a document base contains images of documents associated with metadata, the so-called document structures. The document structure represents the structure of the corresponding document at a given time, as well as information expressing how this structure has been

produced. It has been designed to be extensible, in particular to allow users to define their own document structures according to their needs. In this way, it may be considered as a step towards better reusability.

- *An extensible pool of processings*: Providing the ability to (re)use document processings from various (remote) libraries is another important topic of the DocMining project. The platform makes them interoperate by providing a common integration framework based on the notion of contract, defined between each processing and the document structure.
- *An extensible set of visualization/manipulation user interfaces*: All the users' interactions with the document are defined according to the same plug-in approach. Any user interface connected to the platform may then access and manipulate the whole document structure or part of it.
- *An extensible set of scenarios*: An interpretation task is represented by a scenario expressing the user's strategic knowledge, *i.e.* the parameters of the processings, and the objects and data manipulated by the different processings.

Interactions between these four components are supervised by a controller, which is responsible for running scenarios, updating the document structure, simulating processings, and so on. Its main task consists in controlling how processings and scenarios access the document structure. It selects the required document elements to be processed at each step of a scenario, and updates the global document structure after each processing. When a processing is run, the controller transmits the required document elements and parameters to the processing. On the one hand, using a controller guarantees that all changes made to a document structure are recorded and then may be utilized as a basis for building new scenarios. On the other hand, as the running of the scenario is fully managed by the controller, this ensures that a processing has no direct access to the document structure. This consequently implies that the contract of a processing must specify what parts of the document structure the processing has to access.

Concerning processing design, our contract-based approach has the advantage of being implementation independent, as users can change processing implementations without modifying the contract. The approach also allows users to share a common "parameter schema". Since the contract of each processing is defined in a normalized way, all the platform users communicate with processings in the same way, especially to provide them with their parameter values.

3 The Platform Implementation

The platform implementation is based on a Java/XML architecture and relies on four major components:

- The PSI Library integrates different research work of the PSI Laboratory. It includes chains of processings using statistical and/or structural approaches [6] and it is divided into three sub-libraries dedicated to image processing (filtering, binarization, mathematical morphology, blobs coloring, feature extraction, etc.), classification (similarity search, model reconstruction, etc.), and XML data management. These libraries are written in different languages (Java, C/C++, as well as XSLT stylesheets or XML-QL scripts) and can be run on Windows or Linux operating systems.
- The Qgar software system[2] is developed by the same-named project at LORIA [9]. It is aimed at the design of document analysis applications and includes three parts. QgarLib is a library of C++ classes implementing basic graphics analysis and recognition methods. QgarApps is an applicative layer, including high-level applications (binarizations, edge detection, text-graphics separation, thick-thin separation, vectorization, etc.). Applications may be designed and run using an interactive graphical interface called QgarGUI, that incorporates data manipulation and display capabilities. Any application, either designed apart from Qgar or not, may be integrated into the system by simply wrapping it with a predefined C++

[2] http://www.qgar.org/

class, without recompiling any part of the system itself. The whole system is written in C++ and includes about 50,000 lines of code.

- xmillum (XML illuminator) is a framework for cooperative and interactive analysis of documents developed by the DIVA group at the University of Fribourg [14]. It is based on the philosophy that a document recognition system does not work in a fully automatic way, but cooperates with the user to incrementally learn from its feedback and to adapt itself to different document properties[3]. This approach implies that xmillum offers complex visualization and editing capabilities. Moreover, xmillum is independent from a particular document recognition system, as it does not require any specific language. There is a unique constraint: the original data must be represented in XML. It is subsequently transformed into the xmillum specific structure using an XSLT stylesheet. This specific structure contains:

  – objects, that define how to visualize a data element,
  – styles, that specify the rendering properties of the objects,
  – handlers, that encapsulate the interactions with the objects (such as labelling, merging or splitting objects),
  – selectors, that allow the interaction with a group of objects and indicate to handlers which objects are involved in an action,
  – tools, that are specific modules extending xmillum capabilities (such as multiple views of a document),
  – layers, that are the data containers.

  The xmillum editor interprets this structure, but does not contain any implementation and uses external modules to visualize or edit data. There are four types of modules, displayables (which encapsulate the visualization of objects), handlers, selectors, and tools, that make xmillum freely extensible.

- The ImTrAc package, developed by the GRI Lab at France Télécom R&D Lannion, provides a processing engine, to control process running, and a scenario engine, to control scenario running, as well as tools to integrate new processings in the system and to create scenarios.

The following describes the three major data structures that are used in the implementation (cf. Fig. 1): documents, processings, and scenarios.

3.1 Documents

The DocMining architecture is based on a document-centered approach. A document is represented by an XML tree constructed according to an XML schema. Basic elements are graphical objects defined by their type (`Binary Image`, `Connected Component`, `Text Block`...), their source (the document they come from), and their location in the image. We did not try to elaborate a complete predefined taxonomy of possible types: Platform users can define their own graphical object types, when necessary. A graphical object includes intrinsic data describing the way the object is physically represented, using an XML schema defined from basic data types such as `Freeman Chain`, `Feature Vector`, and so on. Just as previously, a user can define his own data types if necessary.

However, a document is more than a simple description in terms of graphical objects and data. Its structure also contains information (name, parameters...) about processings that have been applied to the document and that have yielded the objects. Objects included in the document structure are visualized using xmillum, thanks to XSLT stylesheets that define which objects may be visualized, how they are visualized, and how events involving objects (mouse clicks, for example) are handled. Each object is associated to a Java class that performs the rendering.

[3]The so-called “CIDRE” philosophy, standing for Cooperative and Interactive Document Reverse Engineering. It is supported by the Swiss National Fund for Scientific Research, code 2000-059356.99-1

3.2 Processings

As previously said, a processing has no direct access to the document structure and cannot modify it unless a so-called contract, defined according to an XML schema, has been established with the document. This contract describes the processing behaviour: The way the processing modifies the XML document structure (by adding, removing, or updating nodes), the kind of graphical objects it produces, and parameters that do not require access to the document structure. The objects that a processing may modify or access are defined by specifying a "trigger" node (that enables the execution of the corresponding processing) and "updated" nodes (that are modified by the processing). This means that the node that launches a processing is not necessarily the one that is modified by the processing. This distinction gives more flexibility to the design of processings, as a user may associate conditions with launching nodes (to require the presence of a particular kind of object, for example) and updated nodes. At the present time, various separate applications from the QgarApps and PSI libraries can be run by the Docmining platform.

3.3 Scenarios

In order to achieve interpretation tasks, users can interactively construct scenarios, that are defined as structured combinations of document processings. There are two ways of creating a scenario. One is based on the contracts of the processings. As objects input and output by processings are specified in the corresponding contracts, it is possible to determine which processings can feed a given processing and then to combine processings.

The other way relies on a xmillum component that has been especially developed for the DocMining platform. It provides means to interact with the processing engine (ImTrAc) and to visualize the structure of the document. For each object of a document, the engine is able to supply the list of processings that may be applied. Once the user has chosen a processing, the engine supplies the parameter list of the processing to the user, so as to get the corresponding values and then to be able to launch the processing. When the processing terminates, ImTrAc updates the document structure and the user can then interact with the newly created objects. Each user action on the document is recorded in the scenario, that may later be applied to another document. In fact, each step of a scenario acts as a trigger and includes an XPath expression describing the way the required objects have to be extracted from the document.

4 A Complete Scenario for Mixed Text/Graphics Documents

The DocMining platform has already been used to produce scenarios dedicated to page segmentation evaluation and to production of ground-truth data sets. The modular architecture of the platform allows the definition of applications concerning various domains: The scenario that is presented here in detail performs the analysis of documents with a mixed text-graphics content. Its aim is the separation of the graphical part from the textual part. Text-related processings (block segmentation, OCR...) are then applied to the textual part, and graphics-related processings (vectorization, thick-thin separation) are applied to the graphical part. Such a scenario may be used for different purposes:

- It may become part of a more general document interpretation system.
- It may be used to evaluate the robustness of an algorithm in case of noisy input images (leading to text-graphics separation errors).
- It may be used as a benchmarking tool: When a developer implements a particular step of the scenario, he may run the different available processings to evaluate the efficiency of his implementation.

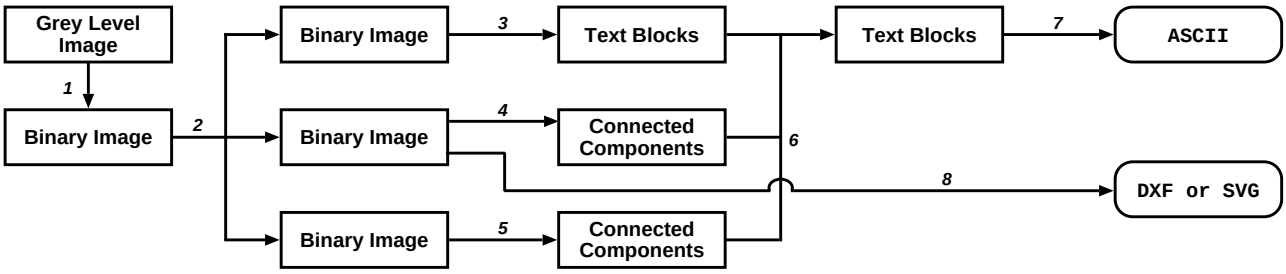


Figure 2: An overview of the scenario for text-graphics separation.

4.1 Scenario Overview

The scenario we want to build can be quickly describe as follows:

- Separate the graphical layer from the textual layer.
- On the textual layer, perform a segmentation into text blocks, and then apply an OCR process on the resulting blocks to produce ASCII texts.
- On the graphical layer, perform a vectorization to get graphical primitives, represented in SVG or DXF format.

The efficiency of such a scenario deeply depends on the efficiency of the text-graphics separation step. If some of the image components are mislabelled or not labelled during this step, further processings cannot be correctly performed. It is therefore necessary to add an extra step dedicated to the correction of labelling errors. The steps of the scenario are finally ordered as follows:

1. Binarize the image.
2. Perform text-graphics separation.
3. Perform segmentation into text blocks on the textual layer image.
4. Perform segmentation into connected components on the graphical layer image.
5. Perform segmentation into connected components on the parts that have not been classified as graphics or text.
6. Correct text-graphics separation errors by a cross-examination of the results of these three segmentations.
7. Perform OCR on the text blocks of the resulting textual layer image.
8. Perform vectorization on the resulting graphical layer image.

Figure 2 shows the corresponding steps of this scenario and the objects manipulated during its running. Such a scenario differs from classical sequential processing chains, that do not include switches like those of steps 2 and 6.

4.2 The Processings

The main processings available to run the scenario are described below:

- *Binarization*. It may be performed using one of four possible processings. The PSI library implements two standards algorithms for automatic computation of binarization thresholds, that have been utilized in a segmentation system for gray level images of cadastral maps [17]: Otsu's method is histogram-based [20], whereas Kittler's method is clustering-based [16]. The QgarApps library supplies an adaptive algorithm proposed by Trier and Taxt [11], with some minor adaptations [25]. The last processing implements a raw method: the binarization threshold is given by the user.

a b c d

Figure 3: Running the scenario: Initial image (a), graphical (b) and textual (c) layers resulting from the text-graphics separation, final image segmented into text blocks (d).

- *Text-graphics separation.* It is based on a method proposed by Fletcher and Kasturi [12] and has been implemented by the Qgar Project, with an extra absolute threshold for the size of a text component [25].
- *Text block segmentation.* At the present time, two processings are available: one is based on a classical top-down approach, the other on an hybrid approach [13]. The latter works by detecting pieces of text lines in small overlapping columns. These pieces are then merged in by a bottom-up algorithm to form complete text lines and blocks of text lines.
- *Connected component segmentation.* It is provided by the PSI image processing (sub-)library, that includes some applications based on a standard blob coloring algorithm [1]: image labelling [7], component analysis (surface, perimeter, position, etc.) [7], and occlusion extraction [6].
- *Correction of text-graphics separation errors.* The purpose of this step is to adequately assign mislabelled or unlabelled connected components of the graphical layer to the textual layer. When a connected component of the graphical layer is located inside a text block of the textual layer, it is assigned to the textual layer. The corresponding bounding box in the graphical layer is also transfered to the textual layer.

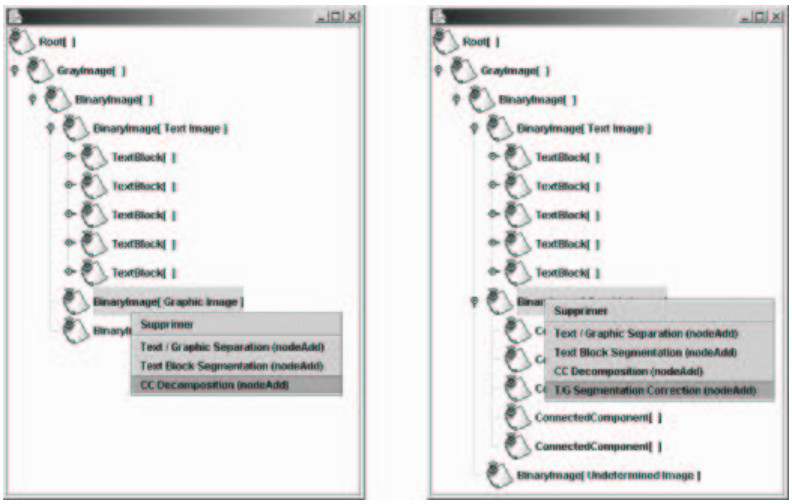
4.3 Results

The execution of the scenario is partly illustrated by figure 3. From an initial gray level image (cf. Fig. 3.a), a text-graphics separation method (step 2) provides three new images representing the graphical layer (cf. Fig. 3.b), the textual layer (cf. Fig. 3.c), and the layer of unclassified parts (not shown here). The cross-examination of the text-blocks extracted from the textual layer and of the connected components extracted from the two other layers (step 6) allows a complete identification of the text blocks present in the initial image (cf. Fig. 3.d). OCR (step 7) and vectorization (step 8) tasks are then respectively applied to the text blocks and to the graphical layer to get a full segmentation of the initial image into graphical and textual primitives...

5 The Construction of the Scenario

The construction of a scenario is performed with an interactive tool called SCI (Scenario Construction Interface), that has been developed to drive the ImTrAc controller. The construction relies on a structural combination of processings belonging to the processing pool. As explained in section 3.3, the combination of processings has been made possible by associating each processing with a contract describing its behaviour. Thanks to this very notion of contract, the elaboration of a scenario does not require any processing to be run.

Figure 4 shows screenshots displaying a partial view of a document tree structure: The popup menu shows authorized processings for the currently selected node (*i.e.* the activation node). We can see that the correction of text-graphics separation errors is not possible (the corresponding processing does not appear in the popup



a b

Figure 4: Coherence checking: The correction of text-graphics separation errors (b) is possible if and only if a connected component segmentation has been performed before (a).

menu of Fig. 4.a) as long as a connected component segmentation has not been performed before (cf. Fig. 4.b: This time, the processing appears in the popup menu). This implies that the coherence of a scenario is always validated during the construction itself of the scenario.

Figure 5 shows the contract defined for the processing that performs the corrections of text-graphics separation errors. The noticeable aspects of the contract, as explained in section 3.2, can be seen: The service (tag `<service/>`), the produced objects (tag `<produced_object/>`), and the handled objects (tag `<xpath_exp/>`).

The contract is not just an XML translation of a classical configuration file. In fact, it includes "processing instructions" as XPath expressions. For example, the eighth line (in bold face) is such an expression and indicates which elements the processing needs. The ImTrAc engine interprets the Xpath expression, extracts the resulting element set from the document and transmits it to the processing.

The contract approach thus allows the delegation of part of the processing, and especially of data preparation for the ImTrac engine. Coupled with the strength of Xpath expressions, this approach adds flexibility to the platform architecture, as it allows the definition of processings that:

- may be triggered from any particular node,
- may modify any nodes of the structure,
- may take any node of the structure as parameters with very precise selection criteria.

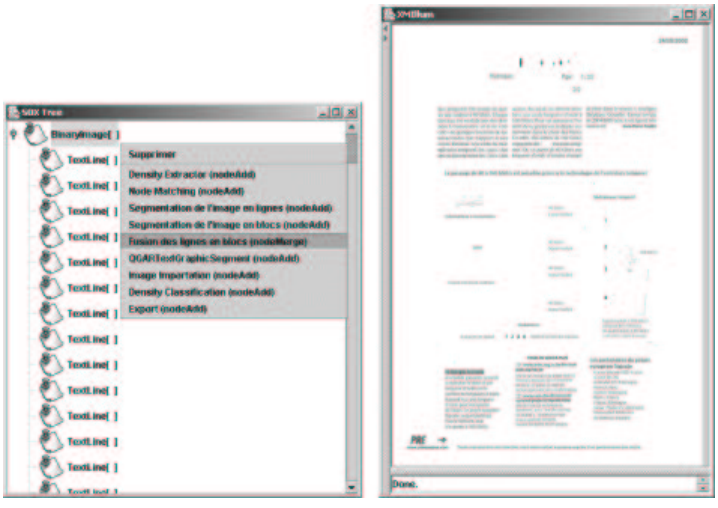
The document structure can be visualized and manipulated with an XSLT script that has been designed for xmillum and that declares a user interface. This interface has been specifically developed to be integrated as an xmillum tool. It drives the ImTrAc engine and applies the selected processing on the document structure. The updated document is visualized depending on objects and styles included in the XSLT script. For example, as shown on Figure 6, the processing to merge text lines into text blocks may be interactively triggered (cf. Fig. 6.a) and the resulting blocks are displayed in a separate window (cf. Fig. 6.b).

6 Conclusion

DocMining is not an interpretation system but a framework. It is a first step towards the definition of a flexible and reusable system to be used as a basis to develop applications related to specific domains or to design

```
<?xml version="1.0" encoding="UTF-8"?>
<process_property class_name="docmining.sox.extern.OrphanMerger">
    <service name="nodeAdd">
       <handled_object>
          <xpath_exp>./object_doc[@type="ConnectedComponent"]</xpath_exp>
          <process_config>
             <param type="Input" name="TextImage" support="ObjectDoc"
              param_value="//object_doc[@name="TextImage"]" />
             <param type="ParamIn" name="Orphans" ObjectDoc"
              param_value="./object_doc[@type="ConnectedComponent"]"/>
             <param type="ParamIn" name="ThreshFactor" support="Data" param_value="2"/>
          </process_config>
          <produced_object>
             <object_doc type="TextBlock">
                <object_doc type="ConnectedComponent">
                   <object_info type_list='yes'/>
                </object_doc>
                   <object_info type_list='yes'/>
                </object_doc>
          </produced_object>
       </handled_object>
    </service>
 </process_property>
```

Figure 5: The contract of the processing that performs the corrections of the text-graphics separation errors.



a b

Figure 6: Screenshots from the user interface integrated into xmillum: Activation of the processing to merge text lines into block lines (a) and resulting blocks (b).

an interpretation system able to run complex scenarios requiring to combine processings possibly coming from different sources (*e.g.* research projects). Among its main advantages, it lets users free in their choices about the information structure manipulated by the system as well as about the implementation of the processings. These ideas has been successfully experimented and validated using scenarios to segment documents into textual and graphical parts.

Acknowledgements

The authors wish to especially thank Oliver Hitz and Rolf Ingold (DIUF Lab, Université de Fribourg), Jean-Marc Ogier (L3I Lab, Université de La Rochelle), Florent Le Dret (Université de La Rochelle), Jan Rendek (Qgar Project, LORIA, Nancy), Yves Lecourtier and Eric Trupin (PSI Lab, Université de Rouen), as well as all the other participants in the DocMining Consortium for their contribution to the work described by this paper.

References

[1] A.K. Aggarwal and A.V. Kulkarni. A sequential approach to the extraction of shape features. *Computer Graphics and Image Processing*, 6(6):538–557, 1977.

[2] J.-F. Arias, C.P. Lai, S. Surya, R. Kasturi, and A.K. Chhabra. Interpretation of telephone system manhole drawings. *Pattern Recognition Letters*, 16(1):355–359, 1995.

[3] H.S. Baird. Anatomy of a versatile page reader. *Proceedings of the IEEE, Special Issue on OCR*, 80(7):1059–1065, 1992.

[4] L. Boatto, V. Consorti, M. Del Buono, S. Di Zenzo, V. Eramo, A. Esposito, F. Melcarne, M. Meucci, A. Morelli, M. Mosciatti, S. Scarci, and M. Tucci. An interpretation system for land register maps. *IEEE Computer Magazine*, 25(7):25–33, 1992.

[5] B. Coüasnon. DMOS: A generic document recognition method. application to an automatic generator of musical scores, mathematical formulae and table structures recognition systems. In *Proceedings of 6th International Conference on Document Analysis and Recognition, Seattle (USA)*, pages 215–220, 2001.

[6] M. Delalandre, S. Nicolas, E. Trupin, and J.-M. Ogier. Symbols recognition by global-local structural approaches, based on the scenarios use, and with a XML representation of data. In *Proceedings of 7th International Conference on Document Analysis And Recognition, Edinburgh (Scotland)*, 2003.

[7] M. Delalandre, Y. Saidali, J.-M. Ogier, and E. Trupin. Adaptable vectorisation system based on strategic knowledge and XML representation use. In *Proceedings of 5th IAPR International Workshop on Graphics Recognition, Barcelona (Spain)*, 2003.

[8] A.R. Dengel and B. Klein. smartFIX: A requirements-driven system for document analysis and understanding. In D. Lopresti, J. Hu, and R. Kashi, editors, *Proceedings of 5th IAPR International Workshop on Document Analysis Systems, Princeton (New Jersey, USA)*, volume 2423 of *Lecture Notes in Computer Science*, pages 433–444. Springer-Verlag, Berlin, 2002.

[9] Ph. Dosch, C. Ah-Soon, G. Masini, G. Snchez, and K. Tombre. Design of an integrated environment for the automated analysis of architectural drawings. In S.-W. Lee and Y. Nakano, editors, *Document Analysis Systems: Theory and Practice. Selected papers from 3rd IAPR Workshop on Document Analysis Systems, Nagano (Japan), November 4–6, 1998*, volume 1655 of *Lecture Notes in Computer Science*, pages 295–309. Springer-Verlag, Berlin, 1999.

[10] Ph. Dosch, K. Tombre, C. Ah-Soon, and G. Masini. A complete system for analysis of architectural drawings. *International Journal on Document Analysis and Recognition*, 3(2):102–116, 2000.

[11] Ø. Due Trier and T. Taxt. Improvement of "integrated function algorithm" for binarization of document images. *Pattern Recognition Letters*, 16(3):277–283, 1995.

[12] L.A. Fletcher and R. Kasturi. A robust algorithm for text string separation from mixed text/graphics images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):910–918, 1988.

[13] N. Gorski, V. Anisimov, E. Augustin, O. Baret, and S. Maximov. Industrial bank check processing: The A2iA CheckReader. *International Journal on Document Analysis and Recognition*, 3(4):196–206, 2001.

[14] O. Hitz, L. Robadey, and R. Ingold. An architecture for editing document recognition results using XML. In *Proceedings of 4th IAPR International Workshop on Document Analysis Systems, Rio de Janeiro (Brazil)*, pages 385–396, 2000.

[15] S. H. Kim and J. H. Kim. Automatic input of logic diagrams by recognizing loop-symbols and rectilinear connections. *International Journal of Pattern Recognition and Artificial Intelligence*, 8(5):1113–1129, 1994.

[16] J. Kittler and J. Illingworth. Minimum error thresholding. *Pattern Recognition*, 19(1):41–47, 1986.

[17] A. Lassaulzais, R. Mullot, J. Gardes, and Y. Lecourtier. Segmentation d'infrastructures de réseau téléphonique. In *Colloque International Francophone sur l'Écrit et le Document, Québec (Canada)*, pages 188–197, 1998.

[18] D. Niyogi, S.N. Srihari, and V. Govindaraju. Analysis of printed forms. In H. Bunke and P.S.P. Wang, editors, *Handbook of character recognition and document image analysis*, pages 485–502. World Scientific, 1997.

[19] A. Okazaki, T. Kondo, K. Mori, S. Tsunekawa, and E. Kawamoto. An automatic circuit diagram reader with loop-structure-based symbol recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3):331–341, 1988.

[20] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979.

[21] B. Pasternak. *Adaptierbares Kernsystem zur Interpretation von Zeichnungen*. Dissertation zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften (Dr. rer. nat.), Universität Hamburg, 1996.

[22] Y. Saidali, S. Adam, J.-M. Ogier, E. Trupin, and J. Labiche. Knowledge representation and acquisition for engineering document analysis. In *Proceedings of 5th IAPR International Workshop on Graphics Recognition, Barcelona (Spain)*, 2003.

[23] H. Samet and A. Soffer. MAGELLAN: Map Acquisition of GEographic Labels by Legend ANalysis. *International Journal on Document Analysis and Recognition*, 1(2):89–101, 1998.

[24] J.H. Shamilian, H.S. Baird, and T.L. Wood. A retargetable table reader. In *Proceedings of 4th International Conference on Document Analysis and Recognition, Ulm (Germany)*, pages 158–163, 1997.

[25] K. Tombre, C. Ah-Soon, Ph. Dosch, A. Habed, and G. Masini. Stable, robust and off-the-shelf methods for graphics recognition. In *Proceedings of 14th International Conference on Pattern Recognition, Brisbane (Australia)*, pages 406–408, 1998.

[26] Y. Yu, A. Samal, and S. C. Seth. A system for recognizing a large class of engineering drawings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(8):868–890, 1997.