Knowledge-based Interpretation of Business Letters[1]

Karl-Hans Bläsius, Beate Grawemeyer,

Isabel John, Norbert Kuhn

Fachbereich Angewandte Informatik

Fachhochschule Trier

Postfach 1826

D-54208 Trier

E-Mail: {blaesius |grawemeyer | john | kuhn}@informatik.fh-trier.de

Abstract

The performance of document analysis systems significantly depends on knowledge about the application domain that can be exploited in the analysis process. Typically, one has to deal with different sources of knowledge like syntactic knowledge, semantic knowledge or strategic knowledge guiding the analysis process.

We present a knowledge based document analysis system based on a knowledge representation language specially designed for document analysis tasks. It allows to model and to interpret structural knowledge about documents and knowledge about the analysis process declaratively in a common framework.

Keywords: Document Analysis, Document Knowledge

1. Introduction

A major problem in office environments is to deal with the huge amount of information that has to be accessible from each office workspace. The exchange of information mainly takes place

[1] This work has been supported by the Stiftung Rheinland-Pfalz für Innovation under grant 8031-38 62 61/228

through electronic or paper documents. To cope with this huge amount of information automatic selection and interpretation processes for these documents are needed. Document analysis systems can help to find the information that is required in a concrete situation. So, document analysis systems or applications become more and more important for integrated office automation systems.

One task that can be tackled by the use of document analysis systems is to group together documents into office procedures. The user can access documents of a concrete procedure by searching such documents which fulfil a set of predicates. These predicates can either refer to content portions of the documents, e.g. those containing the string ”Fachhochschule“, or to attributes attached to a document, e.g. those containing the string ”Fachhochschule“ in the address block. The difference between these two search patterns is that the latter one requires for a logical analysis of the documents under consideration. Accessing documents in office procedures often requires to have documents tagged with a certain type, e.g. tags like order or invoice. Document analysis systems can be used to produce this information almost automatically, either for incoming or for outgoing documents of an office. Thus, a larger amount of information can be attached to documents providing a more detailed structure in the storage of documents. By that, document analysis systems are a valuable tool for setting up an enterprise information system.

However, the question arises why document analysis systems are not widely used in today’s offices. One of the reasons for that may be that analysis tasks are rather domain specific and therefore, one cannot expect a general purpose system. Consequently, most existing document analysis systems are restricted to relative small domains. To adapt an existing system to a new domain is often as time consuming as developing a new system from scratch.

To our belief, this effort can be significantly reduced when certain design guidelines are considered. For us, this has led to a knowledge based formalism to describe documents. It should allow to separate general analysis knowledge form domain specific one. While the former knowledge can be reused in different applications only the latter one has to be reconstructed when moving from one application to another one.

For a new application one has to encode knowledge about the new domain. Usually this includes to describe the contents of the documents, i.e. which logical structure underlies the documents, where these (logical) parts can usually be found or which textual information can be used to identify parts. This is similar to other formalisms for representing documents, e.g. the international standards ODA (Open Document Architecture, [ISO8613]) and SGML (Standard Generalised Markup Language, [ISO8879]).

However, for guiding the analysis process additional knowledge is necessary: one has to specify which knowledge should be used in which situation. Often, this knowledge is encoded directly into the analysis algorithm. We prefer to model the ”How to analyse“ explicitly by what we call strategic knowledge.

In the rest of this paper we present our initial application domain and try to motivate the concepts underlying our document analysis system. This is done in chapter 2. In chapter 3 we describe the knowledge representation language we have designed for that purpose. Chapter 4 presents some experimental results and we finish our paper with some concluding remarks in chapter 5.

2. The Application Domain

At the Fachhochschule Trier we want develop document analysis systems which can be adopted to new applications rather easily. Up to now we have essentially treated the domain of business letters and implemented our system called WINDOK (Knowledge based Interpretation of Documents). Tasks to be solved by our system are

- the classification of the document type
- the determination of relevant parts of a document, like the sender or the recipient and their constituents.

For analysing these documents we model layout knowledge and logical knowledge. Layout knowledge is related to the physical structure of documents. It concerns knowledge like ”The recipient is almost always in the upper left quarter of the letter” or ”An invoice usually contains an invoice table and items”. Logical knowledge regards the content or the meaning of the physical parts like ”The recipient of an invoice must be a customer (and must be found in the customer database)” or ”A letter is usually signed by the sender”.

This knowledge can be used for different analysis tasks for a class of documents. A major task for analysis processes is to attach logical parts to layout objects. Other tasks can be to identify only the address block(s) in a text or to find the total sum in an invoice. Therefore, we need to express relationships between layout and logical knowledge. Sometimes this is expressed by adding knowledge about relative positions of parts within a document or about typical textual contents but it can also be expressed through (mathematical) relations.

The possibility to specify complex relations between parts of documents (e.g. the name in the address part equals the name in the salutations of a letter) is an essential requirement for knowledge representation in our approach. Such relations can hardly be handled by very common knowledge representation frameworks, like frames [Minsky 74] or terminological logics, like KL-ONE based languages [Brachman&Schmolze 85]. The Problems occurring with the use of standard knowledge-representation Formalism and Standard Inference Mechanisms are:

- inadequacy of representation of documents because of the relation between physical structure (Layout) and semantics (Logic) of the Document.
- inefficiency of inference because document analysis specific strategic knowledge cannot be expressed

In other words: we do not mean that no existing, more or less general knowledge representation formalism could be used for document analysis purposes. However, we think that document analysis is a rather specific task where more efficient inference mechanisms can be designed when a special framework is used.

System Environment

Figure 1 illustrates the environment for the use of the system.

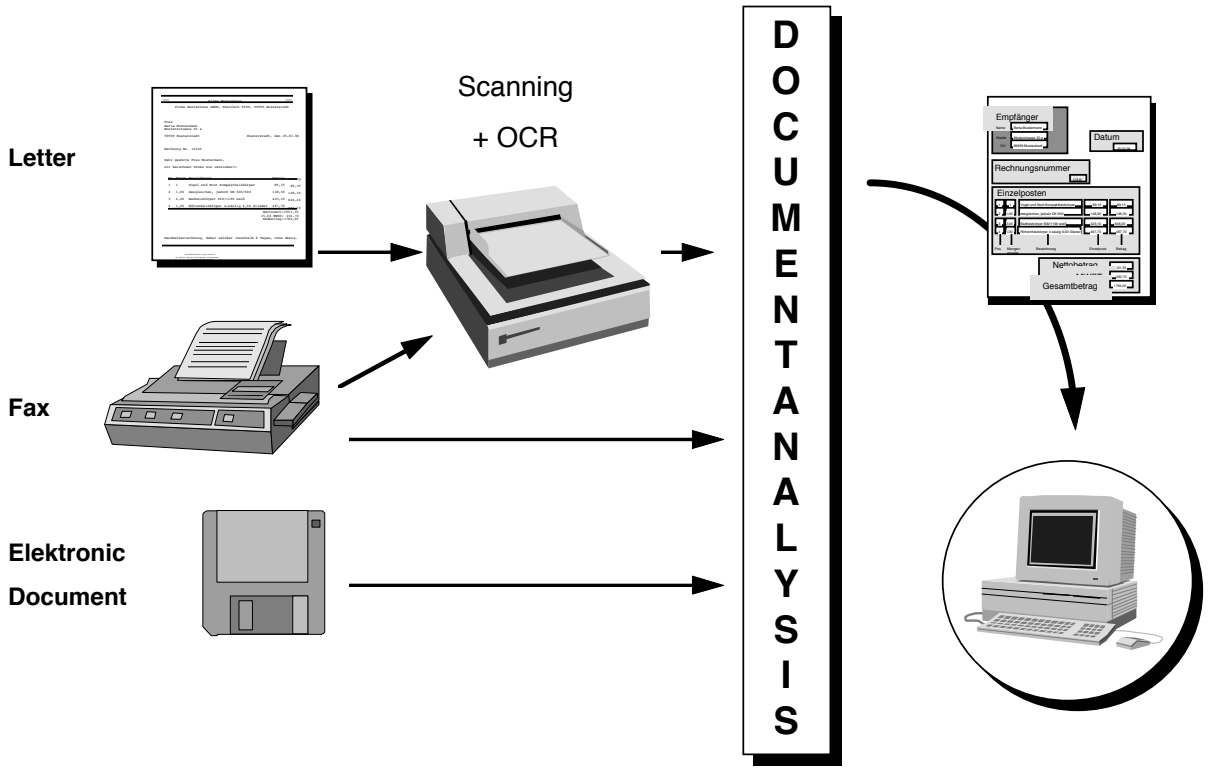

Figure 1. The environment of our document analysis system

System input may be an ASCII file or a printed (paper or fax) document. If a printed document has to be analysed, it is scanned (if necessary) and a commercial OCR-system is used to get ASCII text, which may be enriched with geometric information. This is input to the analysis process.

The WINDOK system consists of the components depicted in *Figure 2*.

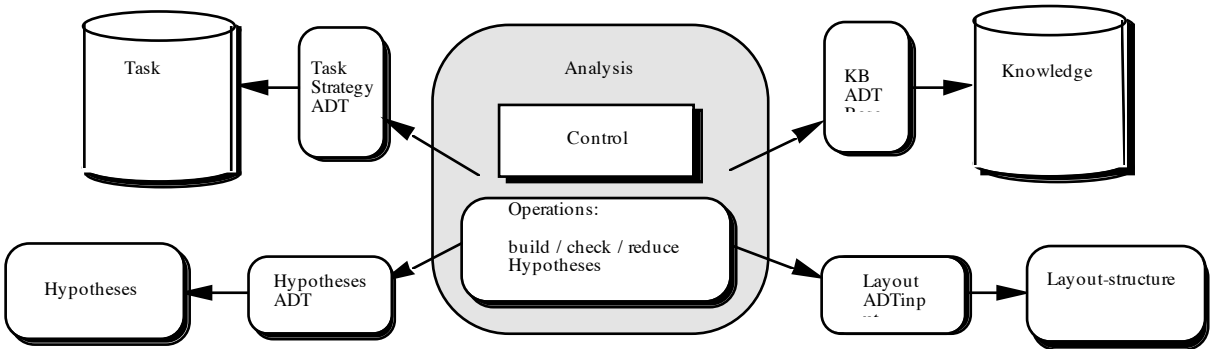

Figure 2. The main components of the WINDOK system

These components are described below.

Analysis Component

The main task of the analysis component is to build and to settle hypotheses about the meaning of the elements of the input document. The analysis component is divided into two main submodules: analysis-control and analysis-operators. The analysis-operators perform basic operations like building hypotheses for the meaning of certain parts of the document, or checking or rejecting certain hypotheses. Analysis-control guides the whole analysis process. That means according to the strategy definitions this component decides which operators are to be applied at which phase of the analysis. For that purpose the analysis-control component uses the information in the strategy definitions, which contain domain dependent knowledge about proper strategies for analysing certain parts of the input document.

The analysis-component has access to the following other components of the system:

- task and strategy definition
- layout structure of the input document
- knowledge base
- hypotheses

Access to these components is only allowed by certain selection-, creation- or modification-functions. That means, the data structures are realised as ”Abstract-Data-Types” (ADT). In the first three cases access is only permitted to get some information (read-only access), these components are not changed in any way by the analysis operations, so this is the static part of the analysis. The access to the hypothesis component is performed in both directions, the hypotheses are created and deleted dynamically during analysis. Analysis-operators need information about the current state of interpretation and produce new understanding about the content of the input document, which is represented in the hypotheses component. By that, hypotheses are extended or refined until a final state is reached where the given tasks are solved, or no further conclusions are possible.

Task and Strategy Definition

The task definition specifies the definite task, i.e. it contains information, whether the type of the documents to be treated is known or whether this type has to be determined. Furthermore the task definition specifies which parts of the documents are to be searched for.

Strategy definitions may be specified for any class of documents or their parts and should contain domain dependent heuristic information, which is used by the analysis-component. Such heuristics may concern the order in which certain parts are to be searched for, which generic properties are to be considered first, or in which order hypotheses should be built, checked or reduced.

Layout Structure

The layout structure contains the result of pre-processing, building a special representation of the input document, consisting of text blocks. Each text block may contain several lines which are built up by words. Words, lines and blocks may be enriched by information about their geometric position on the input document.

Knowledge Base

The knowledge base contains the information of the typical content and structure of documents of a certain class like *letter* or *invoice*. These classes are described declaratively, including parts typically occurring in such documents, as well as relations between these parts. In order to be able to represent such information adequately, a special knowledge representation language has been designed, which is described in section 3.

The knowledge base is used by the analysis component to interpret the input information, i.e. objects of the input document (layout objects) like words, lines or text blocks are related to generic concepts or classes of the knowledge base. By that, the meaning of the layout objects is determined.

Hypotheses

The hypotheses component contains a description of the current state of analysis. For certain parts which are to be searched for or to be analysed, alternatives of interpretation are stored together with probability values. These intermediate solutions are refined by the analysis component until a terminating state is reached, representing the final solution. So, in object-oriented terminology, the hypotheses are partially filled instances of the frame templates which have to be completely filled during analysis.

3. Knowledge Representation

In this section we describe the knowledge representation language in more detail. With this language we can express logic, layout and strategic knowledge. To express strategic knowledge we use the defstrategy and deftask definition but we will not go into further detail for that. With a different strategy and task definition, different document analysis problems can be formulated and solved with more or less the same or similar knowledge bases.

We expect from knowledge representation language that it allows declarative and object-centered descriptions of analysis applications. The knowledge-base should be easy modifiable and good to understand. It should be possible to describe aggregation of objects, uncertain and vague knowledge. Different kinds of knowledge like layout, logic and strategic knowledge should be easy to express as well as relations between objects and parts of (other) objects. In the following sections we describe our solution to achieve these requirements.

For a declarative and object-centered knowledge representation, a frame-based language or a semantic net like representation could be chosen for example. We decided for a frame-based approach because of the built-in inheritance concepts and the possibility for integration of other kinds of concepts like predicate logic. In our language the standard frame concept is extended by descriptions of parts (part-of hierarchies), uncertainty and relations.

A description of an object in our document analysis language is built from the following (optional) elements:

- a name for the object (mandatory)
- the superframes of the object
- the parts of the object

- attributes of the object and its parts
- relations between parts

With these features a definition of a frame invoice can look like shown in figure 3.

```
(defframe Invoice
  (superframes business_letter)
  (parts  (recipient_of_invoice      (frame recipiet))
          (number_of_invoice         (frame number_of_invoice)
          (invoice_table             (frame invoice_table))))
```

Figure 3. An invoice frame

That means an invoice is a business letter with parts recipient of invoice, number of invoice and an invoice table . The structure of a recipient of an invoice is described in the frame recipient.

This frame corresponds to a semantic net like notation as follows (description of attributes see below).

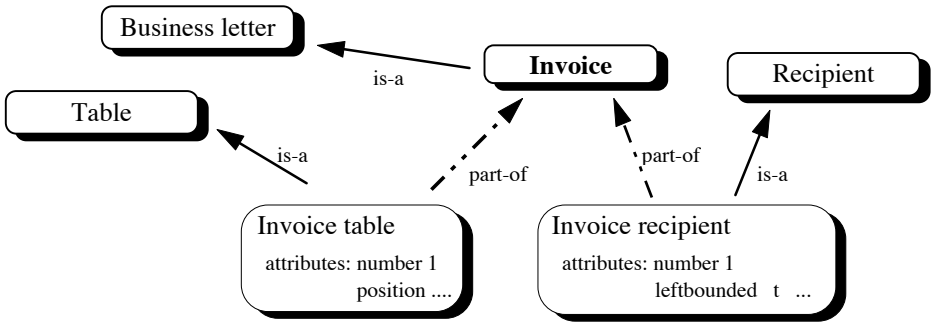

Figure 4. Semantic net representation of frame invoice

All frames or part frames can have parts again, so there is an aggregation (= part) hierarchy in addition to the superframes (= is-a) hierarchy which contains all objects from document down to word or character.

To express the knowledge which is needed during the analysis process we have integrated attributes with several annotations into the frames. Predefined annotations are:

- type (e.g. integer, boolean, real, string....)
- value, used for fixed values like page-width

- range, used to restrict the domain of values (enumeration or interval)
- relevance, for relevance expressions (see below)
- compute-function, used to determine a value for instantiation and testing

For the expressions of uncertainty we use certainty factors [Shortliffe&Buchanan 75] in relevance annotations of attributes. Annotations can have a measure of belief and a measure of disbelief which get reckoned up during analysis. The frame for an invoice table can be described as follows:

```
(defframe invoice_table
   (number                    (type integer)
                              (value 1))
   (parts  (headings          (frame heading)
                    (number   (type integer)
                              (value 1))
                    (position_first_line (type integer)
                              (range          (20 25))
                              (relevance                ((20 22) 0.8 0.0)
                                                        ((23 25) 0.4 0.0)))
                    (number_of_words          (type integer)
                              (range          (5 7))
                              (relevance      (5    0.7 0.0)
                                              ((6 7) 0.9 0.0))))
           (item              (frame item)
                    (number                   (type integer)
                                              (value 1))
                     (position_first_line(type integer)
                              (range          (21 26))
                              (relevance      ((21 23) 0.8 0.0)
                                        ((24 26) 0.4 0.0))))
           (amount            (frame amount) .....)
           (sum               (frame sum)...)))
```

Figure 5. An invoice table frame

This means that there is only one invoice table with part headings where the relative position of the first line is between 20 and 25. For other values of position, no hypothesis can be built and so no relevances are given. The compute-function or other annotations of the attribute may be defined somewhere else in the knowledge base, either in a superframe of this frame or in an additional defattribute construct for global definition of annotations. The other parts and their attributes are defined similar.

The relations between frames and parts mentioned until now regard only is-a and part-of relations. A description of all relationships between possible objects that could be useful (or helpful) for document analysis tasks. In our knowledge representation language it is possible to

model arbitrary relations between frames and their parts or frames and other frames. The relations can either be used to reduce hypotheses or to build new hypotheses with instances which fulfil the constraints given by the relations.

Relations are, like attributes, defined within the frame definitions. But as they normally have an arity greater than one, they refer to several parts or the frame itself. In Figure 6 is an example of some of the relations of the frame whose attributes we have already shown above.

The first relation describes that the headings, must begin above the items the third relation describes that the part item number of the heading must be located above the item number part of any single item. Here, parts of parts of frames are needed to properly express the relation.

```
(defframe Invoice_table ...
   .(attributes and parts see above)
      (relations          (< (headings position_first_line)
                             (item position_first_line))
                  (< (item position_first_line)
                             (amount position_first_line))
                          (over (headings number_of_item)
                                (item single_item number_of_item))
                  (over (headings description)
                             (item single_item description)))
```

Figure 6. Relations of an invoice table frame

With these relations all instances (words, lines...) that fulfil the relations can be found and can serve as good hypotheses for analysis. Ideally, when modelling of the relations is done well, there are only a few candidates to be checked further, in strategy definitions concerning other frames or other relations.

4. Experimental Results

So far we have tried to solve the following problems:

- Analysis of ASCII texts with addresses of companies and authorities in text flow
- Classification of document types (Invoice, Order, Offer)

- Analysis of Invoices with address, table and items.

As an example we want to show here the analysis of an invoice (similar work was done by [Köppen&al 96]).The document image is pre-processed by a common OCR-software providing text and layout information which is then transformed into our internal representation.

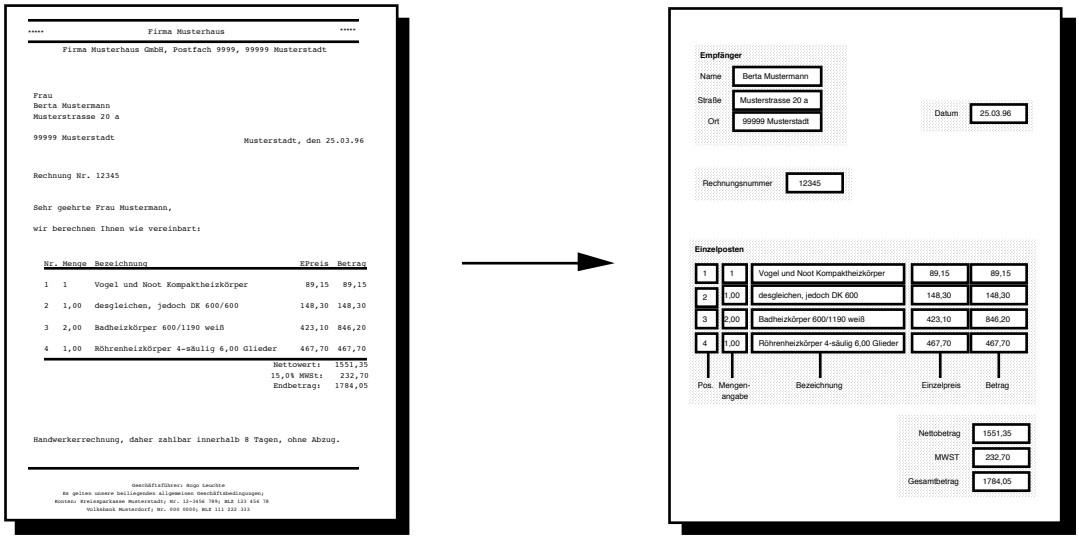

Figure 7. Example of processing an invoice

The analysis of address date and invoice table is done according to the strategies as described in chapter 2 and assisted by knowledge about their structure given in the knowledge base. Address, date and table are analysed following the strategies which do not rely on a certain invoice template but are flexible in order to analyse all kinds of invoices. The results we obtained are shown in figure 7. This data obtained through Document Analysis can be put into a data-base and can accomplish the company memory.

We tested our System with a test sample of about 50 Business Letters (mostly invoices) and obtained the results shown in figure 8.

%	correct	incorrect	not analyzed
recipient	69,57	2,17	28,26
date	83,72	0	16,28
single items	71,80	2,56	25,64
sum	84,09	6,82	9,09

Figure 8. Test Results

For the analysis of different problems, tasks and strategies have to be changed and the knowledge base has to be adapted to the application domain.

5. Conclusion and Outlook

We consider knowledge representation as an essential and central activity for the development of document analysis systems. We model knowledge about the structure and the content of documents of the actual application domain and strategic knowledge to guide the analysis process itself.

Our specialised knowledge representation formalism allows for a specification of all these kinds of knowledge. Another issue of this approach is that descriptions of documents can be understood and maintained also by non specialist users. Furthermore, existing descriptions can be reused more easily in new application domains.

The system is implemented in Allegro Common Lisp and runs on different platforms, like Apple Macintosh, Sun and IBM PC.

Currently, we are working on several domains (Letters, Invoices, Fax-Messages, …) to gain some insight in the effort that has to be spent when a system for a new domain is set up. Our first experiences are encouraging. Furthermore, we work on extensions of the knowledge representation language which are necessary to cope with other classes of documents. Furthermore, some enhancements concerning uncertainty will be made by implementing uncertainty formalisms which fit better for our document analysis tasks.

6. Acknowledgements

All authors wish to thank the DFKI GmbH for giving them the opportunity to participate in the OMEGA and the PASCAL 2000 projects of the Document Analysis group. This has inspired our work and helped us to achieve the results presented in this paper.

7. References

[Bayer 93a] T. Bayer. *Understanding Structured Documents By a Model Based Document Analysis System*. Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR), pages 565 - 568, Tsukuba Science City, Japan, 1993

[Bayer 93b] T.Bayer. *Ein modellgestütztes Analysesystem zum Bildverstehen strukturierter Dokumente*. Diski 44, Infix, St.Augustin, Germany, 1993. *In German*

[Bayer&al 94] T.A. Bayer, U. Bohnacher, H. Mogg-Schneider. *InfoPortLab - An Experimental Document Analysis System*. Proceedings of the IAPR-Workshop on Document Analysis Systems, Kaiserslautern, Germany, 1994

[Bläsius&Hönes 95] K.-H. Bläsius, F. Hönes. *A Mechanism for Generating Image Analysis Algorithms*. Proceedings of the Workshops KI'95, Bielefeld, Germany, 1995

[Brachmann&Schmolze 85] R.J. Brachmann, J.G. Schmolze. *An Overview of the KL-ONE Knowledge Representation System*. Cognitive Science 9, pages 171-216, 1985

[Dengel 89] A. Dengel. *Automatische Visuelle Klassifikation von Dokumenten*. Doctoral Thesis, Universität Stuttgart, 1989. *In German*

[Dengel&al 94] A. Dengel, R. Bleisinger, F. Fein, R. Hoch, F. Hönes, and M. Malburg. *OfficeMAID - A System for Office Mail Analysis, Interpretation and Delivery*. Proceedings of First International Workshop on Document Analysis Systems (DAS'94), pages 253-275, Kaiserslautern, Germany, October 18-20 1994

[Dengel&Dubiel 95] A. Dengel and F. Dubiel. *Clustering and Classification of Document Structure- A Machine Learning Approach*. Proceedings of the Int. Conference of Document Analysis and Recognition (ICDAR '95), Montreal, Canada, 1995

[ISO8879] ISO 8879. *Information Processing- Text and Office Systems - Standard Generalised Markup Language (SGML)*. 1986

[ISO8613] ISO 8613, *Information Processing- Text and Office Systems -Office Document Architecture (ODA) and Interchange Format*. vol. I-III, parts 1-8, 1988

[Köppen&al 96] M.Köppen, D.Waldöstl, B. Nickolay. *A System for the automated evaluation of invoices*. Proceedings International WS on Document Analysis Systems (DAS '96), pages 3-21, Philadelphia, 1996

[Minsky 74] M.Minsky. *A Framework for Representing Knowledge.* MIT AI Lab. AIM 306, Cambridge, 1974

[Pasternak 94] B. Pasternak. *Processing Imprecise and Structural Distorted Line Drawings by an Adaptable Drawing Interpretation Kernel*. Proceeding of the IAPR-Workshop on Document Analysis Systems, pages 349 - 365, Kaiserslautern, Germany, 1994

[Pasternak & Neumann 93] B. Pasternak, B. Neumann. *Adaptable Drawing Interpretation Using Object-Oriented and Constraint Based Graphic Specification*. Proceedings of the 2nd Int. Conference on Document Analysis and Recognition (ICDAR), pages 359 - 364, Tsukuba Science City, Japan, 1993

[Shortliffe&Buchanan 75] E.H Shortliffe, B.G Buchanan.. *A Model for Inexact Reasoning in Medicine*. Mathematical Biosciences 23, pages 351 -379, 1975

[Wenzel&al 96] C. Wenzel, S. Baumann, T. Jäger. *Document Classification by Voting of Competitive Approaches*. Proceedings of International Workshop on Document Analysis Systems (DAS '96), pages 352-374, Philadelphia, 1996