An Approach to Context-driven Document Analysis and Understanding

Claudia Wenzel, Heiko Maus

German Research Center for Artificial Intelligence (DFKI)
P.O. Box 2080, 67608 Kaiserslautern/Germany
E-mail: wenzel@dfki.de, maus@dfki.de

Abstract. Most of today‘s document analysis systems are employed for special-purpose tasks and carry their knowledge directly within their algorithms. However, certain office environments demand for more flexible applications of document analysis and understanding (DAU) techniques having a broader application space and being subject to frequent changes while still requiring high precision.
Our knowledge-centered approach to that represents not only comparatively static knowledge about document properties and analysis results within the same declarative formalism, but also includes the analysis task and the current system context within this formalism. This allows an easy definition of analysis tasks and an efficient and accurate analysis by using expectations about incoming documents as context information.
Our approach has been implemented within the VOPR[1] system which gains the required context from a commercial workflow management system (WfMS) by constant exchanges of expectations and analysis tasks.

1 Introduction

Within the last ten years, document analysis systems have made their arrival in different application areas. The requirements which emerge from these applications are quite heterogeneous because of distinct layout structures (e.g., for bank cheques, music), different analysis goals and varying number and quality of the document images. Thus, special-purpose systems are the best solution for most applications.

However, there is a justification for more general document analysis systems. Such systems are not omnipotent but they are not limited to exactly one analysis task and one type of documents. Typical application areas are insurance claims in general, accounts for travel expenses, business letters and so on. The system architecture of a system dealing with such a general class of documents is characterized by a declarative knowledge representation providing document properties but sometimes strategic knowledge about the analysis procedure. We refer to such systems as *knowledge-based DAU systems*.

For example, Bayer [1] has developed the FRESCO formalism for the definition of knowledge about document components and about analysis algorithms. Documents and their parts along with layout and logical properties are represented by concepts (for generic documents resp. parts) and instances (for concrete images). Analysis components

[1] VOPR is an acronym for the Virtual Office PRototype.

are described in a taxonomical hierarchy by defining application properties. During analysis, a global scheduler starts analysis components which generate instances for document parts. Fuzzy-based inference mechanisms combine instances to valid document instances. A similar approach is described by Bläsius et al. [2], but they use the dempster-shafer approach instead of fuzzy sets for the propagation of uncertainty values. Lam [3] describes a frame-based approach especially for logical labeling. Further representation formalisms for generic document analysis purposes are based on predicate logic [4] and production rules [5].

In the industrial application of document analysis systems, one success factor counts for every analysis system: accuracy. When trying to combine high accuracy with a general application area, different solutions come into play; e.g., the employment of many special purpose analysis algorithms and an explicit scheduler within one common system. However, our approach to accuracy is a little bit more intuitive from the application view: We include semantic context information available within the system environment to restrict the quantity of valid analysis results as early and as far as possible.

Within literature, the usage of application context is not very common. A lot of analysis systems use context coded as heuristics within the analysis procedure, but this allows only a strict analysis without any means for flexibility. The only context usage described being easily accessible and exchangeable is data available within databases. It is typically used for the verification of OCR results, e.g., legal account numbers and bank codes within bank cheques. One approach which combines a declarative description of document properties with context from databases is described by Bleisinger et al. [6]. Their IntelliDoc system has been developed for the analysis of structured business letters. Document properties concerning layout, logic, and content are entered within a graphical user interface by using a specific syntax. Within the same interface, connections to valid entries in databases can be established. During analysis, document knowledge is used by a speech recognizing component and a syntactic pattern-matcher.

Our VOPR approach is able to deal with different analysis tasks by taking its context information from databases and a surrounding workflow management system (WfMS). Such a WfMS automates the handling of administrative business processes such as purchasing processes. The main contribution of the VOPR system to workflow management is to bridge the media gap for document-based business processes, e.g., by directly assigning incoming business letters to the workflow instance waiting for this particular letter and making contained information accessible.

The resulting integration of a DAU system into a WfMS-based business environment is displayed in Fig. 1. Data flow indicated by arrows starts with a clerk's input to the WfMS and with new incoming documents. New incoming documents (scanned and in an electronic representation) are directly transferred to the DAU system. The WfMS supplies the DAU system with context from instantiations of workflow definitions. This kind of corporate knowledge is annotated by entries of a separate corporate database (e.g., postal addresses of suppliers). Thus, the DAU system receives relevant expectations about expected documents from the WfMS and enriches them by retrieving information from the corporate database. When the DAU system is invoked, incoming documents are analysed by different components explicitly selected according to the current task. Having finished DAU, the analysis control returns its results to the waiting

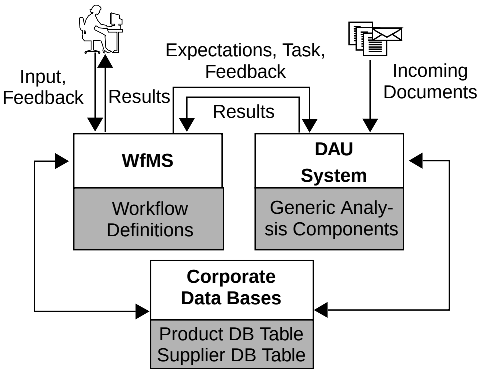


Fig. 1 Integration of Workflow Management and Document Analysis

workflow instance which presents them to the user. She corrects wrong DAU results which may later-on lead to new document knowledge derived by learning components.

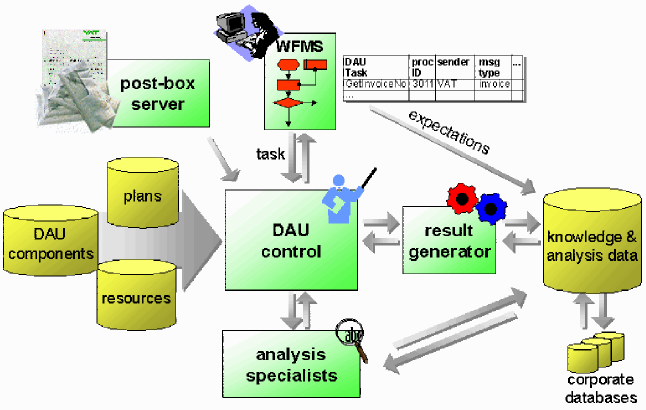
The reminder of this paper describes the VOPR system in more detail. The next chapter clarifies our context notion within the application domain of a purchasing process. Then, all main components are explained: Chapter 3 explains the context collection within a WfMS and its delivery to the DAU system. The integration of context within the knowledge base of the DAU system is subject to chapter 4, while chapter 5 discusses analysis algorithms and their execution by our analysis control. The paper is completed by a qualitative evaluation in chapter 6 and some concluding remarks in chapter 7.

2 Context and its effects on a working system architecture

Besides modeling static document properties, we also include the current application context of the system environment. It is represented by so-called *context units* which consist of an expectation about an incoming document, the information need of the business process in terms of DAU tasks, and administrative data for WfMS integration.

More detailed, a context unit first describes the content and meaning of an expected document and its relationship to the referring business process by stating all previously known facts about a document (e.g., the document‘s message type or the product list of an invoice, where these products have already been ordered in the business process). Furthermore, references to preceeding documents are included, such as information from the corresponding inquiry in case of an expected offer. Moreover, a context unit expresses the information need of a business process by including analysis tasks which describe the required information. In our scenario two tasks were identified: The first one is called *process determination* and is given inherently in the business letter domain because incoming documents relate to business processes to which they have to be assigned. The second one is the task of *information extraction* which requests specific information from the document necessary for further processing in the business process.

Thus, context information from business processes forms a valuable source for a knowledge-based DAU which draws some effects on the VOPR system architecture (see Fig. 2). The architecture mainly displays components for a typical knowledge-



DAU Task	proc ID	sender	msg type	...
GetInvoiceNo	3011	VAT	invoice	
...				

Fig. 2 System architecture of the VOPR system

based DAU: Central part is the DAU control which communicates with the post-box server (responsible for delivering the documents) and the WfMS which automates the underlying business processes. The DAU control uses resources and DAU components to assemble analysis specialists used by task-specific plans to accomplish the tasks. These analysis specialists use knowledge and analysis data within the document knowledge and incorporated databases to fulfill their specific analysis tasks and to store their analysis hypotheses. These hypotheses are input to the result generator which unifies them to one result offered to the requesting workflow. To enable the system to cope with context information, several components have been extended with appropriate interfaces. A WfMS is extended to deliver context to the document knowledge by means of expectations. In addition, DAU components are extended to use these expectations for fulfilling their tasks more efficiently and accurate (e.g., by search space reduction). Also, declarative analysis tasks are introduced to cope with the dynamic environment.

Besides using dynamic context information from business processes, we also model more static context by means of *standard context units* for two purposes: First, they describe documents not expected by a specific running business process such as adverts. Second, they serve as an exception handling for the VOPR system if there is no match between an incoming document and context units from the WfMS or if unexpected documents referring to an existing process such as reminders arrive. Given this default handling for incoming documents, the VOPR system is more than state-of-the-art in the field of inbox processing where documents are routed to workqueues according to predefined keywords (e.g., [6]).

3 Workflow

To deliver context to DAU, we have to access a workflow‘s context. Unfortunately, today‘s WfMS do not have modeling concepts for workflow context (cf. [9]), thus there is no possibility to directly access context as a whole at runtime. However, several sources, e.g., control flow, data flow, application data, or a workflow‘s history [10] exist. To collect all this input, we introduce a database named *context pool* (see Fig. 3)

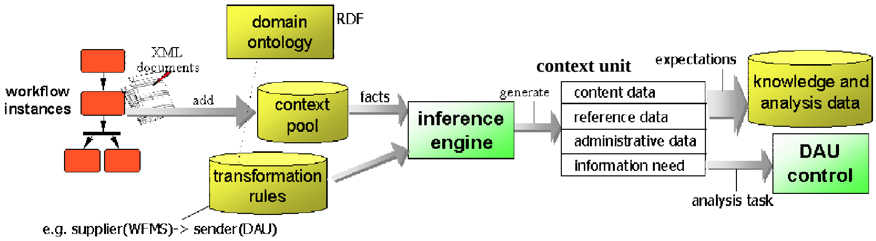


Fig. 3 Generation of context information from WfMS

which guarantees the availability of context when needed. Therein, all valuable context information is collected during workflow execution. Technically, we model some special activities in the workflow definition.

Information within the context pool is semantically described in terms of a domain ontology. We use XML [11] to represent the documents produced within the workflow, e.g., an outgoing order. Furthermore, we modeled the business letter domain as an RDF (Resource Description Framework, www.w3c.org/rdf) schema and used it to semantically enrich XML documents by relating data to this ontology. Due to the possibility to use namespaces in XML, several schemas can be incorporated within a document and used throughout the workflow. This provides a powerful means for semantics in WfMS and for intelligent assistants as mentioned in [12].

In case an event occurs in a workflow instance (e.g., issuing an order) which causes the future arrival of a document (e.g., an invoice), the workflow states an expectation by means of a context unit. Therefore, the workflow invokes an inference engine and hands over some administrative data, the information which caused the event (e.g., the order document), and any additional information need. Given this, the context pool serves as a fact base for the inference step which converts this raw context information into a context unit by using transformation rules.

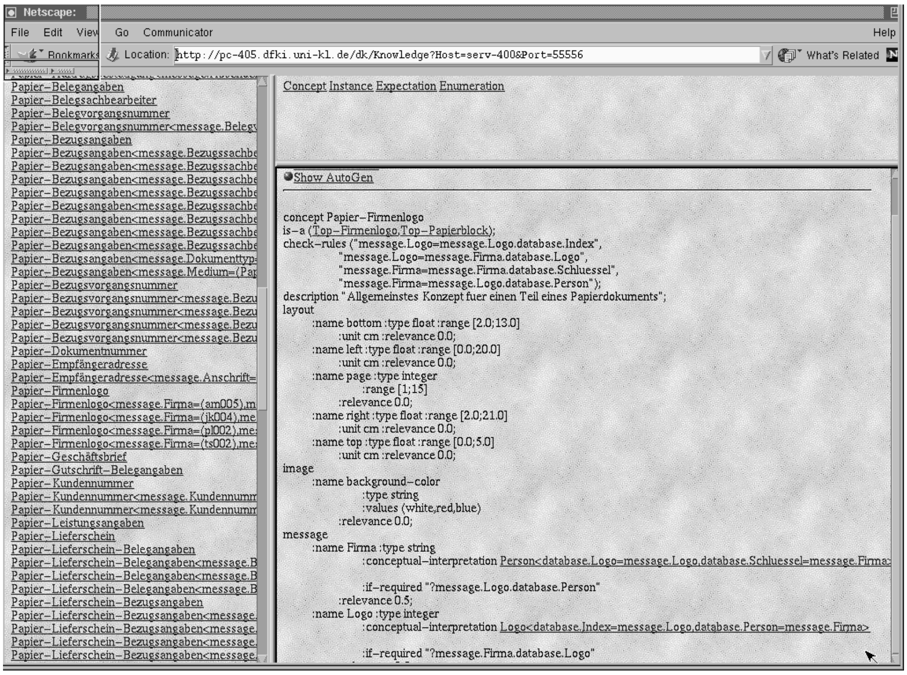
They provide a mapping between the workflow‘s domain ontology and a description of the expected document in terms of the DAU domain ontology (represented by the DAU‘s knowledge base). To derive content and meaning of the expected document, some rules must be domain-specific, e.g., a rule stating that the recipient of an inquiry is to be the sender of the corresponding offer, or a rule assembling information from an available offer to describe potential references within the expected invoice. This information is stored within the content and reference data of the context unit. Other rules generate administrative data such as the calling workflow activity, the workflow‘s instance id and next action. Finally, some rules integrate the information need of the workflow into a context unit by listing data pieces to be extracted in terms of the DAU domain ontology. Next, content and reference data is inserted into the document knowledge as expectation and the information need is placed at the DAU control‘s disposal.

After analysis, a context unit is satisfied by an incoming document matching the given expectation. Hence, the document is assigned to this particular context unit. If there is an additional information need, the DAU performs the task necessary to extract the requested data and stores its results in a designated database. Afterwards, the next action within the workflow instance is performed according to the administrative data.

The next step within the workflow is the verification of the assignment and any additional data. After that, document and data can be processed within the workflow.

4 Representing document knowledge

There is one central repository which stores generic document properties and analysis results called *document knowledge*. Besides storing typical document knowledge and data, some powerful mechanisms have been included to represent context within the same formalism. This enables a direct context access for all analysis components and a semantic exploitation without any format adaptations.



Location: http://pc-405.dfki.uni-kl.de/dk/Knowledge?Host=serv-400&Port=55556

```
Papier–Belegangaben
Papier–Belegsachbearbeiter
Papier–Belegvorgangsnummer
Papier–Belegvorgangsnummer<message.Belegv
Papier–Bezugsangaben
Papier–Bezugsangaben<message.Bezugssachbe
Papier–Bezugsangaben<message.Bezugssachbe
Papier–Bezugsangaben<message.Bezugssachbe
Papier–Bezugsangaben<message.Bezugssachbe
Papier–Bezugsangaben<message.Bezugssachbe
Papier–Bezugsangaben<message.Bezugssachbe
Papier–Bezugsangaben<message.Bezugssachbe
Papier–Bezugsangaben<message.Bezugssachbe
Papier–Bezugsangaben<message.Dokumenttyp=
Papier–Bezugsangaben<message.Medium=(Pap
Papier–Bezugsvorgangsnummer
Papier–Bezugsvorgangsnummer<message.Bezu
Papier–Bezugsvorgangsnummer<message.Bezu
Papier–Bezugsvorgangsnummer<message.Bezu
Papier–Bezugsvorgangsnummer<message.Bezu
Papier–Dokumentnummer
Papier–Empfängeradresse
Papier–Empfängeradresse<message.Anschrift=
Papier–Firmenlogo
Papier–Firmenlogo<message.Firma=(am005),m
Papier–Firmenlogo<message.Firma=(jk004),me
Papier–Firmenlogo<message.Firma=(pl002),me
Papier–Firmenlogo<message.Firma=(ts002),me
Papier–Geschäftsbrief
Papier–Gutschrift–Belegangaben
Papier–Kundennummer
Papier–Kundennummer<message.Kundennumm
Papier–Kundennummer<message.Kundennumm
Papier–Leistungsangaben
Papier–Lieferschein
Papier–Lieferschein–Belegangaben
Papier–Lieferschein–Belegangaben<message.B
Papier–Lieferschein–Belegangaben<message.B
Papier–Lieferschein–Belegangaben<message.B
Papier–Lieferschein–Bezugsangaben
Papier–Lieferschein–Bezugsangaben<message.
Papier–Lieferschein–Bezugsangaben<message.
Papier–Lieferschein–Bezugsangaben<message.
Papier–Lieferschein–Bezugsangaben<message.
```

```
concept Papier–Firmenlogo
is–a (Top–Firmenlogo,Top–Papierblock);
check–rules ("message.Logo=message.Logo.database.Index",
            "message.Logo=message.Firma.database.Logo",
            "message.Firma=message.Firma.database.Schluessel",
            "message.Firma=message.Logo.database.Person");
description "Allgemeinstes Konzept fuer einen Teil eines Papierdokuments";
layout
      :name bottom :type float :range [2.0;13.0]
            :unit cm :relevance 0.0;
      :name left :type float :range [0.0;20.0]
            :unit cm :relevance 0.0;
      :name page :type integer
                 :range [1;15]
            :relevance 0.0;
      :name right :type float :range [2.0;21.0]
            :unit cm :relevance 0.0;
      :name top :type float :range [0.0;5.0]
            :unit cm :relevance 0.0;
image
      :name background–color
                 :type string
                 :values (white,red,blue)
            :relevance 0.0;
message
      :name Firma :type string
                 :conceptual–interpretation Person<database.Logo=message.Logo,database.Schluessel=message.Firma>

                 :if–required "?message.Logo.database.Person"
            :relevance 0.5;
      :name Logo :type integer
                 :conceptual–interpretation Logo<database.Index=message.Logo,database.Person=message.Firma>

                 :if–required "?message.Firma.database.Logo"
```

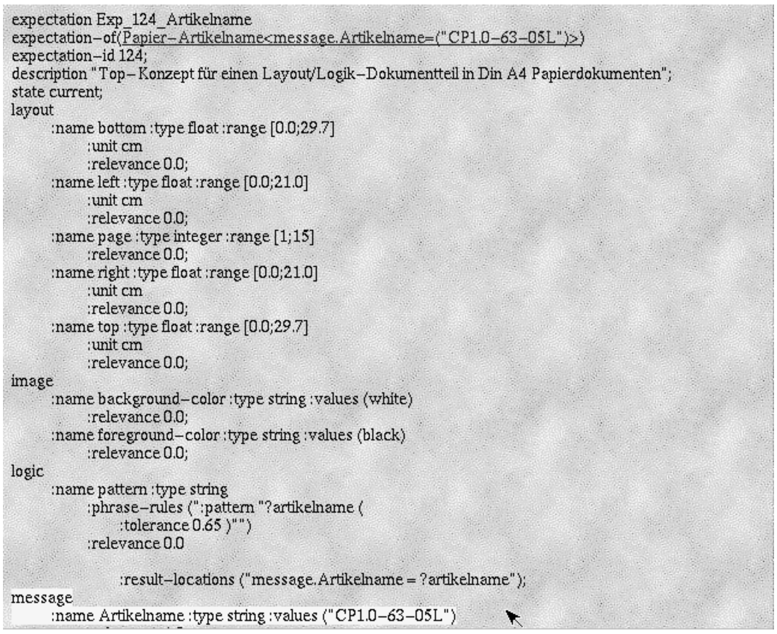
Fig. 4 Screenshot of the document knowledge browser for the concept company logo

Our representation formalism is a frame-based notation where frames represent documents or document parts. Possible relations between frames are specialisations (*is-a*) and aggregations (*parts*). Each frame consists of slots which specify document properties by facets and facet values. Typical contents of document models are well-known in the document analysis community [7], [8]. We distinguish between image (e.g., background-color), layout (e.g., coordinates), content (e.g., language), logic (e.g., syntactic text patterns) and message (all relevant semantic information) properties of a document. There are also different types of facets which may be used for slots, namely for representing uncertainty (`:certainty`), values and value types (`:value`, `:range`, `:unit`, ...), related thesaurus information (`:thesaurus-name`, `:thesaurus-value`, ...), frequencies and relevances. Besides this, we also have some special constructs such as check-rules defining restrictions between slots and frames to ensure consistency.

The formalism is used for storing knowledge gained within a separate learning step automatically (which is not subject of this paper), for retrieving general properties and analysis results and - most important - for combining single analysis results for document parts to consistent overall results for a whole image under consideration.

Generic document properties are represented in frames called *concepts*. An example for a company logo in Fig. 4 shows a screenshot of our knowledge browser: In the upper right area, you can select which kinds of frames to investigate (in our case, "Concept" has been chosen). On the left-hand side, you see all frames of the chosen kind currently defined (e.g., the first one "Papier-Belegangaben" means paper-record data). The lower right area contains the definition of the frame chosen (Papier-Firmenlogo means a company logo on paper). The check-rules in this example state that the company the logo belongs to must be consistent with the document‘s sender.

Within the message slotgroup of the example, you see how context from databases is introduced: Within the slot Logo, the facet `:conceptual-interpretation` defines a link to another frame (named Logo) representing the structure of a database table. Such a frame type is called *dataface* and relates slots within the document knowledge directly to columns of a database table. This allows transparency when accessing databases via document knowledge since all database operations are hidden within the functionality of the document knowledge.



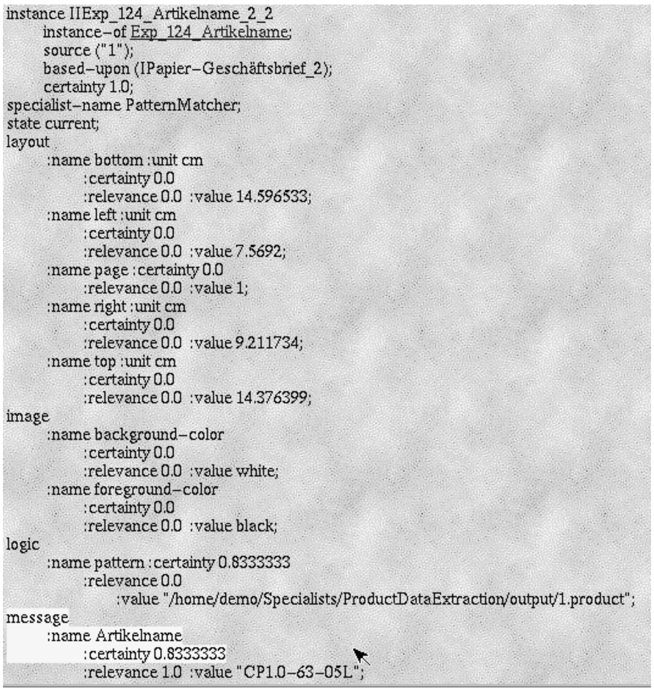
```
expectation Exp_124_Artikelname
expectation–of(Papier–Artikelname<message.Artikelname=("CP1.0–63–05L")>)
expectation–id 124;
description "Top–Konzept für einen Layout/Logik–Dokumentteil in Din A4 Papierdokumenten";
state current;
layout
        :name bottom :type float :range [0.0;29.7]
                :unit cm
                :relevance 0.0;
        :name left :type float :range [0.0;21.0]
                :unit cm
                :relevance 0.0;
        :name page :type integer :range [1;15]
                :relevance 0.0;
        :name right :type float :range [0.0;21.0]
                :unit cm
                :relevance 0.0;
        :name top :type float :range [0.0;29.7]
                :unit cm
                :relevance 0.0;
image
        :name background–color :type string :values (white)
                :relevance 0.0;
        :name foreground–color :type string :values (black)
                :relevance 0.0;
logic
        :name pattern :type string
                :phrase–rules (":pattern "?artikelname (
                        :tolerance 0.65 )"")
                :relevance 0.0

                        :result–locations ("message.Artikelname = ?artikelname");
message
        :name Artikelname :type string :values ("CP1.0–63–05L")
```

Fig. 5 Screenshot of the document knowledge browser for an expectation of an article name

The second kind of context information relevant for DAU has already been mentioned: Context in form of possible contents of expected documents. From the document knowledge point-of-view, these expectations are restrictions of more general document types or their parts, respectively. Therefore, incoming expectations are entered as specialisations of concepts already defined (e.g., an invoice for a printer of the company HewlettPackard is a specialisation of an invoice of the company HewlettPackard). The example given in Fig. 5 shows an expectation for a document part dealing with the name of an article ordered. This frame type is called *expectation*. The only entry in which this expectation differs from the more general concept Artikelname (not shown here) is the name of the article in the last row "CP1.0-63-05L" which is the name of a specific cooling element.

Now imagine, an analysis component has generated a result for the analysis of this document part. This result is stored within a frame called *instance* as shown in Fig. 6. Within the source slot, a unique number for the document image analysed is given (here number 1), the name of the analysis component is pattern-matcher, and within the message slot Artikelname (last few rows), we see that the cooling element CP... has been matched with a certainty of 0.833.



```
instance IIExp_124_Artikelname_2_2
      instance–of Exp_124_Artikelname;
      source ("1");
      based–upon (IPapier–Geschäftsbrief_2);
      certainty 1.0;
specialist–name PatternMatcher;
state current;
layout
      :name bottom :unit cm
            :certainty 0.0
            :relevance 0.0  :value 14.596533;
      :name left :unit cm
            :certainty 0.0
            :relevance 0.0  :value 7.5692;
      :name page :certainty 0.0
            :relevance 0.0  :value 1;
      :name right :unit cm
            :certainty 0.0
            :relevance 0.0  :value 9.211734;
      :name top :unit cm
            :certainty 0.0
            :relevance 0.0  :value 14.376399;
image
      :name background–color
            :certainty 0.0
            :relevance 0.0  :value white;
      :name foreground–color
            :certainty 0.0
            :relevance 0.0  :value black;
logic
      :name pattern :certainty 0.8333333
            :relevance 0.0
                  :value "/home/demo/Specialists/ProductDataExtraction/output/1.product";
message
      :name Artikelname
            :certainty 0.8333333
            :relevance 1.0  :value "CP1.0–63–05L";
```

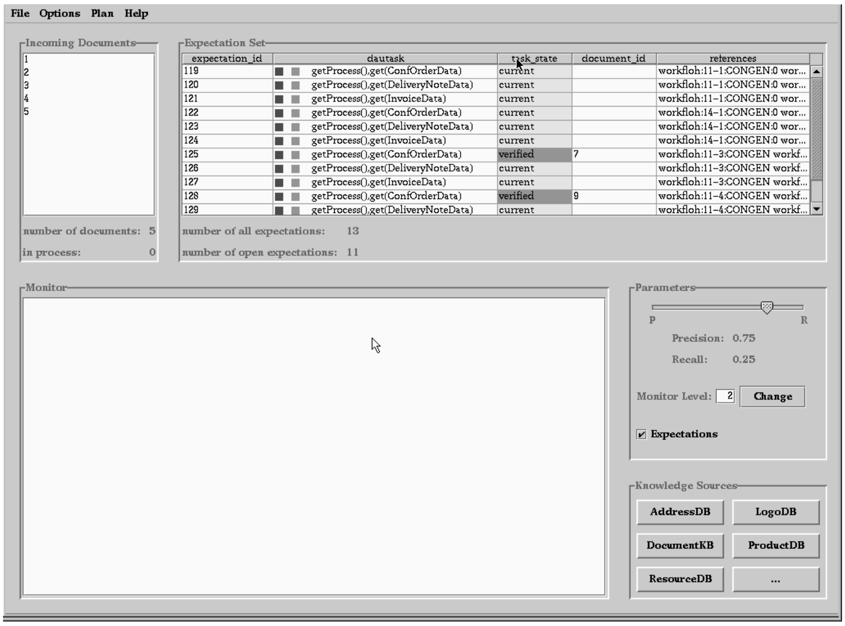
Fig. 6 Screenshot of the document knowledge browser for an instance of an article name

5 Analysis control

The DAU system is triggered in two different situations: In the first situation, the postbox server triggers DAU control with a new incoming document. In such a case, a process assignment has to be accomplished. The DAU control just retrieves the corresponding plan which denotes a sequence of DAU specialists along with pre- and postconditions and alternative paths. With the aid of a corresponding resource file, each specialist can be constructed on the basis of a generic DAU component. Therefore, a resource denotes which specialist extracts which kind of information in a declarative way by using general paths formulated in the document knowledge. In addition, the resource contains necessary parameters, paths within the document knowledge to be investigated, hardware restrictions and so on. Using resources and components, specialists are invoked and controlled according to plans. Having executed the analysis plan, the DAU control transfers the matching expectation id to the workflow management system. If requested, additional extracted information is retrieved from the document knowledge (by inheritance mechanisms) and handed over to the workflow.

In the second situation, DAU control is invoked when the workflow asks for additional information from a document which has already been assigned to a process. Such a new information extraction task is also specified by formulating general paths in terms of the document knowledge. In this case, the analysis control retrieves a general information extraction plan and instantiates it. That means, that all specialists are invoked with a restricted task which is far more efficient.

Our analysis control is visualized by a DAU control post which allows the visualization and inspection of document knowledge, expectations, plans, and document images. Furthermore, parameter settings such as precision or time requirements can be set up here. Fig. 7 shows the starting window of the control post. For each document, the analysis plan can be inspected or started in a trigger mode. In this case, a separate window shows all necessary information (see also Fig. 8).



Incoming Documents: 1, 2, 3, 4, 5

number of documents: 5

in process: 0

expectation_id	dautask	task_state	document_id	references
119	■ ■ getProcess(),get(ConfOrderData)	current		workfloh:11–1:CONGEN:0 wor...
120	■ ■ getProcess(),get(DeliveryNoteData)	current		workfloh:11–1:CONGEN:0 wor...
121	■ ■ getProcess(),get(InvoiceData)	current		workfloh:11–1:CONGEN:0 wor...
122	■ ■ getProcess(),get(ConfOrderData)	current		workfloh:14–1:CONGEN:0 wor...
123	■ ■ getProcess(),get(DeliveryNoteData)	current		workfloh:14–1:CONGEN:0 wor...
124	■ ■ getProcess(),get(InvoiceData)	current		workfloh:14–1:CONGEN:0 wor...
125	■ ■ getProcess(),get(ConfOrderData)	verified	7	workfloh:11–3:CONGEN workf...
126	■ ■ getProcess(),get(DeliveryNoteData)	current		workfloh:11–3:CONGEN workf...
127	■ ■ getProcess(),get(InvoiceData)	current		workfloh:11–3:CONGEN workf...
128	■ ■ getProcess(),get(ConfOrderData)	verified	9	workfloh:11–4:CONGEN workf...
129	■ ■ getProcess(),get(DeliveryNoteData)	current		workfloh:11–4:CONGEN workf...

number of all expectations: 13

number of open expectations: 11

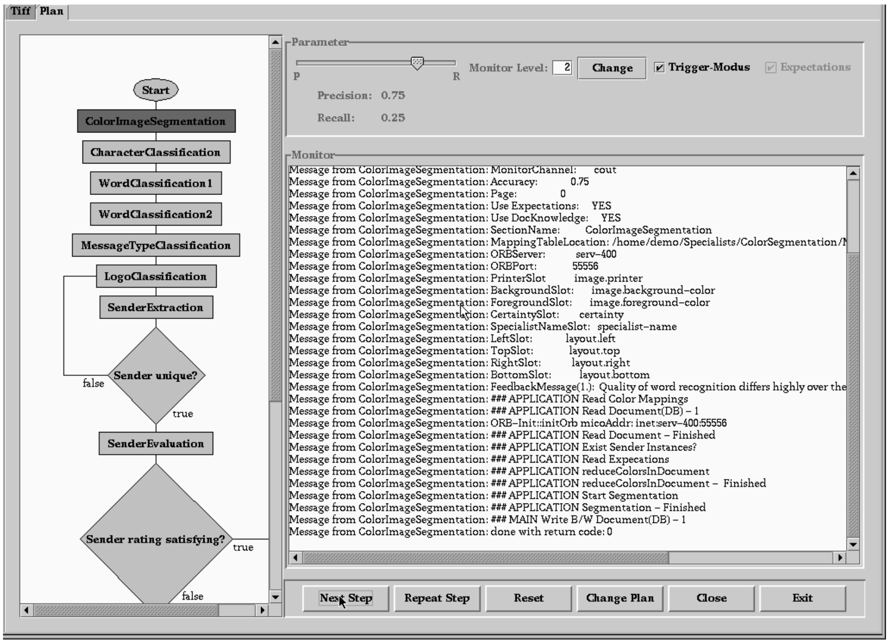
Precision: 0.75

Recall: 0.25

Monitor Level: 2

☑ Expectations

Fig. 7 Screenshot of the analysis control post



Monitor Level: 2
☑ Trigger-Modus
☑ Expectations

Precision: 0.75
Recall: 0.25

```
Message from ColorImageSegmentation: MonitorChannel:     cout
Message from ColorImageSegmentation: Accuracy:         0.75
Message from ColorImageSegmentation: Page:          0
Message from ColorImageSegmentation: Use Expectations:    YES
Message from ColorImageSegmentation: Use DocKnowledge:    YES
Message from ColorImageSegmentation: SectionName:      ColorImageSegmentation
Message from ColorImageSegmentation: MappingTableLocation: /home/demo/Specialists/ColorSegmentation/M
Message from ColorImageSegmentation: ORBServer:        serv–400
Message from ColorImageSegmentation: ORBPort:         55556
Message from ColorImageSegmentation: PrinterSlot        image.printer
Message from ColorImageSegmentation: BackgroundSlot:     image.background–color
Message from ColorImageSegmentation: ForegroundSlot:     image.foreground–color
Message from ColorImageSegmentation: CertaintySlot:      certainty
Message from ColorImageSegmentation: SpecialistNameSlot:  specialist–name
Message from ColorImageSegmentation: LeftSlot:         layout.left
Message from ColorImageSegmentation: TopSlot:          layout.top
Message from ColorImageSegmentation: RightSlot:         layout.right
Message from ColorImageSegmentation: BottomSlot:        layout.bottom
Message from ColorImageSegmentation: FeedbackMessage(1.):  Quality of word recognition differs highly over the
Message from ColorImageSegmentation: ### APPLICATION Read Color Mappings
Message from ColorImageSegmentation: ### APPLICATION Read Document(DB) – 1
Message from ColorImageSegmentation: ORB–Init::initOrb micoAddr: inet:serv–400:55556
Message from ColorImageSegmentation: ### APPLICATION Read Document – Finished
Message from ColorImageSegmentation: ### APPLICATION Exist Sender Instances?
Message from ColorImageSegmentation: ### APPLICATION Read Expecations
Message from ColorImageSegmentation: ### APPLICATION reduceColorsInDocument
Message from ColorImageSegmentation: ### APPLICATION reduceColorsInDocument –  Finished
Message from ColorImageSegmentation: ### APPLICATION Start Segmentation
Message from ColorImageSegmentation: ### APPLICATION Segmentation – Finished
Message from ColorImageSegmentation: ### MAIN Write B/W Document(DB) – 1
Message from ColorImageSegmentation: done with return code: 0
```

Fig. 8 Screenshot of a plan execution

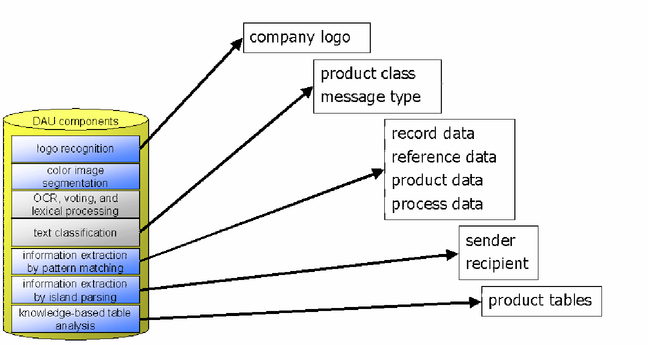


Fig. 9 DAU components (left-hand side) and resulting specialists (right-hand side)

All DAU components which have been integrated into the VOPR system are displayed in Fig. 9. Components which heavily rely on declarative document knowledge may be transformed into different domain- and task-specific specialists. For such components, the resulting specialists currently employed are shown on the right hand side of Fig. 9. Now a short description of each component (see also [13], [14], [15]) follows: **color image segmentation.** Starting with a color reduction step, the component generates a contrast representation which shows significant color differences in adjacent pixels. It is used to construct a color connected component hierarchy on the basis of the single-pass algorithm. Subsequently, scanner errors are reduced by using typical scanner profiles. Finally, the color image is converted into a black/white image.

logo recognition. The logo recognizer is a by-product of the color image segmentation. First, graphic elements in appropriate document regions are selected as logo candidates. With that, the list of valid logos is filtered by comparing simple features (moments, numbers of colors). The shape of the remaining candidates is compared by recursive tree-matching. Output is a valued ranking of possible logos.

text classification. For the textual classification of a document, we employ an inductive rule learner. It learns patterns and boolean expressions on word level during the learning phase and uses fuzzy matching for these expressions in the classification phase. Its output is a valued ranking of matching classes.

OCR, voting, and lexical processing. The outputs of three OCR engines (M/Text, TextBridge, EasyReader) are stored in a complex graph structure. Based on that, our voting component combines character hypotheses by comparing word and line segments and by matching corresponding character graphs. Afterwards, lexical processing matches lists, prefixes, suffixes and regular expressions for valid words against the voting results and calculates confidence measures for the final word hypotheses.

pattern matcher. It allows an error-tolerant, shallow information extraction based on regular expressions for syntactic text patterns. They are processed by combining confidence measures from lexical processing (bottom-up) and from document knowledge (top-down). The component uses similarity measures for words based on morphology (word stems, part-of-speech), semantics (synonyms, hypernyms), geometry, and fonts. It generates results for layout/message slotgroups of the document knowledge.

parser. It accomplishes a deep syntactic analysis for those documents parts which have a strong internal structure. Its kernel is an island parser with a stochastic grammar. The parser generates results for logic and message slotgroups of the document knowledge.

knowledge based table analysis. It analyses tables in documents which are instantiations of previously defined table models. Analysis is based on the proper detection of a table header by different features (geometry, lines, keywords,...). As a result, the table's structure as well as its contents on cell level are extracted.

There is another (non-analysis) component but typically included at a plan‘s end:

result generator. The result generator combines instances produced by other specialists to a final instance for the whole document image. It is mainly a search algorithm with uncertainty propagation (combination is based on a procedure similar to the MYCIN expert system). Search is based on a chart-parsing approach which allows a high flexibility in the search strategy.

Those components which can be transformed into several specialists are applicable in situations with or without expectations. When dealing with expectations, several strategies can be used: The *closed world strategy* restricts the specialists application only to expectations and only results which instantiate expectations are produced. The *well-rounded strategy* allows results which are consistent with expectations while the *comprehensive strategy* allows the generation of both, results based on expectations and results based on more general concepts at the same time. The strategy used influences the number and kind of analysis instances which are input for the result generator. There is no best strategy because this depends on basic assumptions of the surrounding system environment (How many and which unexpected documents may occur beneath those modeled in standard context units?).

6 Qualitative evaluation

Within a real company, hundreds of open expectations at one time are realistic. It is not possible to simulate this within a research environment since the construction of a correct benchmarking environment is too time consuming. One has to simulate one workflow instance for each expectation which must fit to an available paper document, ground-truth data has to be provided and the correct process has to be determined.

Because of this, we tested the system as a whole up to now with 18 expectations at one time. For process determination, we identified sixteen information items which may be at hand in expectations, e.g, sender, message type, process number, total amount. However, when simulating the original processes (from which we had all incoming and outgoing documents), we found out that in a typical process, about ten of these items are explicitly mentioned. At the document side, we tested the system with 12 documents which typically contained about six relevant information items. Process determination with the usage of expectations was always correct. However, when neglecting expectations for analysis, the final analysis hypothesis would have led to a wrong process determination in four cases. Moreover, the usage of expectations shortened the runtime of single specialists (e.g., logo recognition, pattern matcher) to a high amount because of the resulting restrictions of the search space.

We also investigated the impact of expectations to DAU components (e.g., in the following for the pattern matching component) in detail. Therefore, we looked at the reasons for errors within 250 documents (doing a „conventional“ concept-based analysis). Information items analysed were numbers, dates, proper names, and prices. Error rates for these categories ranged from 17-43%, but the usage of expectations can decrease these rates between 20 and 45%. The reason for that is that a lot of errors depended on character recognition problems for single characters (which can be nearly totally repaired by an error-tolerant matching based on expectations).

7 Conclusion

This paper presented a prototypical implementation of a knowledge-based DAU system within an automated business process environment. It uses a generic document knowledge allowing to incorporate different knowledge sources such as corporate databases and context information from WfMS. Thus, the VOPR system enables context-driven document analysis and understanding resulting in faster system runs (e.g., reduced search space, extracting only requested data), higher precision (e.g., by using background knowledge), and flexible analysis tasks. Moreover, the system enables learning capabilities (by incorporating verification results), and is domain and WfMS vendor independent (e.g., generic document knowledge; context pool).

The system accomplishes a tight integration of DAU into WfMS and bridges the gap between paper-based parts of communication and automated business processes. Further benefits for workflow management are the reduction of transport and idle times, less expenses for data capturing, and the automation of a company's inbox interface. The presented solution for context collection which considers current trends in e-commerce (XML) is efficient and unintrusive. The only efforts remain at buildtime by including some activities in the workflow definition for implementing the integration.

Our future work will further evaluate the impact of expectations. We will assess the expenses of efforts for context collection against the revenue in analysis results. Furthermore, we will estimate the amount of minimal context information necessary to achieve an appropriate assignment rate in real-world quantities of documents.

Acknowledgements

Thanks to our (partly former) colleagues Andreas Lauer, Jürgen Lichter, Michael Malburg, Harald Mayer auf'm Hofe and Tino Sarodnik for designing and implementing the VOPR system with us. The project has been funded by the German Federal Ministry for Education and Research under grant number 01 IW 807.

References

1. T. Bayer. *Understanding Structured Text Documents by a Model-Based Document Analysis System.* Second Int. Conf. on Document Analysis and Recognition, Tsukuba Science City, Japan, Oct. 1993, pp. 448-453.
2. K.-H. Bläsius, B. Grawemeyer, I. John, N. Kuhn. *Knowledge-Based Document Analysis.* Fourth Int. Conf. on Document Analysis and Recognition, Ulm, Germany, Aug. 1997.
3. S. W. Lam. *An Adaptive Approach to Document Classification and Understanding*. First workshop on document analysis systems, Kaiserslautern, Germany, 1994, pp. 231-251.
4. F. Esposito, D. Malerba, G. Semeraro, C. D. Antifora, G. de Gennaro. *Information Capture and Semantic Indexing of Digital Libraries through Machine Learning Techniques*, Fourth Int. Conf. on Document Analysis and Recognition, Ulm, Germany, Aug. 1997, pp. 722-727.
5. R. Ingold. *A Document Description Language to Drive Document Analysis,* First Int. Conf. on Document Analysis and Recognition, Saint-Malo, France, Sept./Oct. 1991, pp. 294-301.
6. R. Bleisinger, M. Müller, P. Hartmann, T. Dörstling. *Intelligente Eingangspostverarbeitung mit wissensbasierter Dokumentanalyse*, Wirtschaftsinformatik Vol. 41, 1999, pp. 371-377.
7. G. Nagy. *What does a machine need to know to read a document?* Symposium on Document Analysis and Information Retrieval, Las Vegas, Nevada, 1992, pp. 1-10.
8. S. Srihari. *From pixels to paragraphs: the use of models in text recognition.* Second Int. Conf. on Document Analysis and Recognition, Tsukuba Science City, Japan, Oct. 1993.
9. U. Remus, F. Lehner. *The Role of Process-oriented Enterprise Modeling in Designing Process-oriented Knowledge Management Systems*. 2000 AAAI Spring Symposium, Bringing Knowledge to Business Processes, Stanford, California.
10. S. Jablonsky, Ch. Bussler. Workflow Management. *Modeling Concepts, Architecture and Implementation.* International Thomson Computer Press, 1996.
11. E. R. Harold. *XML Bible*. IDG Books Worldwide, Inc, 1999.
12. A. Abecker, A. Bernardi, H. Maus, M. Sintek, C. Wenzel. *Information Supply for Business Processes: Coupling Workflow with Document Analysis and Information Extraction*. Elsevier's Journal of Knowledge-based Systems, Special Issue on Artificial Intelligence in Knowledge Management, 2000.
13. T. Jäger, A. Pies, A.Weigel. *Global versus Local Combination of OCR Results.* Proc. of the 5th JCIS, Feb./Mar. 2000, Atlantic City, New Jersey, USA.
14. C. Wenzel, M. Malburg. *Lexicon-Driven Information Extraction from a Document Analysis View.* Workshop on Lexicon Driven Information Extraction, Frascati, Italy, July 1997.
15. C. Wenzel, W. Tersteegen. *Precise Table Recognition by Making Use of Reference Tables.* Seong-Whan Lee, Yasuaki Nakano (Eds.), Lecture Notes in Computer Science, Springer Verlag, Vol. 1655, Aug. 1999.