IJDAR (2001) 3: 248–260

International Journal on Document Analysis and Recognition IJDAR
© Springer-Verlag 2001

Leveraging corporate context within knowledge-based 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,maus}@dfki.uni-kl.de

Received June 19, 1999 / Revised November 8, 2000

Abstract. Knowledge-based systems for document analysis and understanding (DAU) are quite useful whenever analysis has to deal with the changing of free-form document types which require different analysis components. In this case, declarative modeling is a good way to achieve flexibility. An important application domain for such systems is the business letter domain. Here, high accuracy and the correct assignment to the right people and the right processes is a crucial success factor. Our solution to this proposes a comprehensive knowledge-centered approach: we model not only comparatively static knowledge concerning document properties and analysis results within the same declarative formalism, but we also include the analysis task and the current context of the system environment within the same formalism. This allows an easy definition of new analysis tasks and also an efficient and accurate analysis by using expectations about incoming documents as context information. The approach described has been implemented within the VOPR[1] system. This DAU system gains the required context information from a commercial workflow management system (WfMS) by constant exchanges of expectations and analysis tasks. Further interaction between these two systems covers the delivery of results from DAU to the WfMS and the delivery of corrected results vice versa.

Key words: Document analysis system – Office processes – Context information – Document knowledge

1 Introduction

The analysis of printed business letters has become a lucrative application domain for document analysis and understanding (DAU) techniques within the last few years. Some commercial systems are available on the market for this purpose, e.g., AscentCapture by Kofax [1], TELEform from Cardiff [2] or Free Form from Captiva [3]. The goal of such systems is to analyse incoming business letters in order to extract all relevant information and to assign them to the right people working within the right processes. This saves inhouse mail distribution and manual indexing efforts.

In order to adapt DAU systems to the changing and specific requirements of distinct companies, most of these commercial systems (according to published details) keep the layout, and logical and textual knowledge about the documents within the current company's domain in a declarative style. We refer to such systems as knowledge-based document analysis and understanding (DAU) systems.

Besides document properties, there is other knowledge available which is useful for analysis and can be kept declarative, such as the analysis task or the corporate context. We argue that using declarative modeling as far as possible leads to several advantages which are especially useful in real-world applications.

To this end, Fig. 1 explains the state of the art and typical way of document analysis for printed business letters on the left-hand side and compares it to our own VOPR approach. Each way starts with incoming mail which is fed to a system for document analysis and understanding. Then, the DAU system analyses the documents according to a specific analysis task. The specification of this analysis task already reveals a difference between the two views: in the traditional view, the analysis task is static and predefined to a special extent; for the goal of process assignment, some keywords have been connected to the names of employees within the company. Such relations might be "Invoices $\rightarrow$ Mrs. Snider" (which means that Mrs. Snider treats all invoices) or "Company Microsoft $\rightarrow$ Mr. Gates" (which means that Mr. Gates treats all letters coming from the company Microsoft). The kinds of keywords to be used are predefined (e.g., keywords must be company names) and are system-internally hardwired to the analysis task (e.g., sender extraction) and to analysis components (e.g., a pattern matcher). Thus, some specific analysis tasks are employed to determine information items which relate

Correspondence to: H. Maus

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

the document in a static way to people within the company. Moreover, a message frame has been attached to each keyword which specifies the kind of information to be extracted from the document for the employee. The message frame itself is also directly related to a specific analysis component which is able to process the frame's syntax. Thus, the overall analysis is accomplished in two steps: in the first step, the matching keyword (and therefore the corresponding employee and the message frame) is determined. In the second step, all information denoted in the message frame is extracted from the letter. Finally, the document, along with its filled message frame, is sent to the corresponding employee. To sum up, a predefined kind of information given by the keyword is used as the only connection between the incoming letter and the system environment within the company.

However, the system environment is more than just a person's inbox. In today's companies, administrative procedures dealing with business letters are often automatized with the aid of workflow management systems (WfMS for short). WfMS get the right data to the right people with the right tools at the right time [4].

Thus, instead of always assigning incoming mail to a mail tool of specific persons, it makes sense to directly attach incoming mail to the corresponding workflow instances. In this case, people involved get the data directly with the right tool at the right place. How to accomplish this is subject of this paper and is shown at the right-hand side of Fig. 1.

Roughly speaking, a *context set* is steadily collected by looking at all open workflow instances (current context units) and by modeling default cases (standard context units). This context set represents all currently possible analysis tasks as well as application context from the workflow instances. The application context states the expectations of the WfMS to future incoming documents such as the document's type, its sender, or its topics. The analysis of incoming letters uses the context set for a goal-directed analysis and attaches letters along with extracted information usually directly to waiting workflow instances. Just in case that an incoming document starts a new process (e.g., a request) or is only related to certain persons, the system attaches letters to people's inboxes or default workflow definitions.

As we will prove later on, this VOPR approach has some crucial advantages over the typical approach:

- The declarative modeling of application context, analysis task and message frames enables an easy adaptation to new kinds of documents and new message frames, but also to new company processes.
- The semantic application context can be used to restrict the search space for some analysis components. This allows for a higher analysis efficiency.
- For the same reason, we also reach a higher accuracy.

The remainder of this paper describes the VOPR system in more detail. We start with a description of related work in the next section. Afterwards, Sect. 3 explains the underlying system architecture. Then, all main components are explained: Sect. 4 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 Sect. 5, while Sect. 6 discusses the expectation-based execution of analysis algorithms by our analysis control. Section 7 details the analysis algorithms employed and Sect. 8 explains their relation to different analysis strategies. The paper is completed by some experimental analysis results in Sect. 9 and some concluding remarks in Sect. 10.

2 Related work

Within the document analysis literature, the usage of application context is not very common. Of course, a lot of systems use context coded as heuristics directly within the analysis procedure, but this allows only a strict analysis and there are no means for exchanging such heuristics. The only context usage described and which is easily accessible and exchangeable is data available within databases. Such data is typically used for the verification of OCR results, e.g., legal account numbers, customer names, and bank codes within bank cheques.

However, there are commercial as well as research systems which deal with the analysis of business letters. One commercial system for which results have been published is described by Bleisinger et al. [5]. Their IntelliDoc system combines a declarative description of document properties with context from databases. It has been developed for the analysis of more or less structured business letters. Document properties concerning layout, logic, and content are entered within a graphical user interface by using a system-specific syntax. Within the same interface, connections to valid entries in databases can be established. Unfortunately, details about this integration are not mentioned. During analysis, document knowledge is used by a speech recognizing component and a syntactic pattern-matcher.

Another research approach which is exploited commercially is the FRESCO formalism (frame representation of structured documents) developed by Thomas Bayer [6]. It has been applied to German and English business letters. It allows the definition of knowledge about document components as well as defining knowledge about analysis algorithms. Documents and their parts along with layout and logical properties are represented by concepts (for generic documents and/or parts) and instances (for concrete images). Analysis components are described in a taxonomical hierarchy by defining application properties. During analysis, a global scheduler chooses analysis components. These components generate instances for document parts and fuzzy-based inference mechanisms combine instances to valid document instances.

In a similar way, our own previous work within the Omega project, see, for example, [7] has been put on the market. Our research prototype contained a toolbox of task-specific analysis specialists for image preprocessing of gray-scale documents, OCR, voting, morphological analysis, logical labeling, document classification, and so on. These specialists communicated with the aid of a blackboard architecture and worked on the domain of

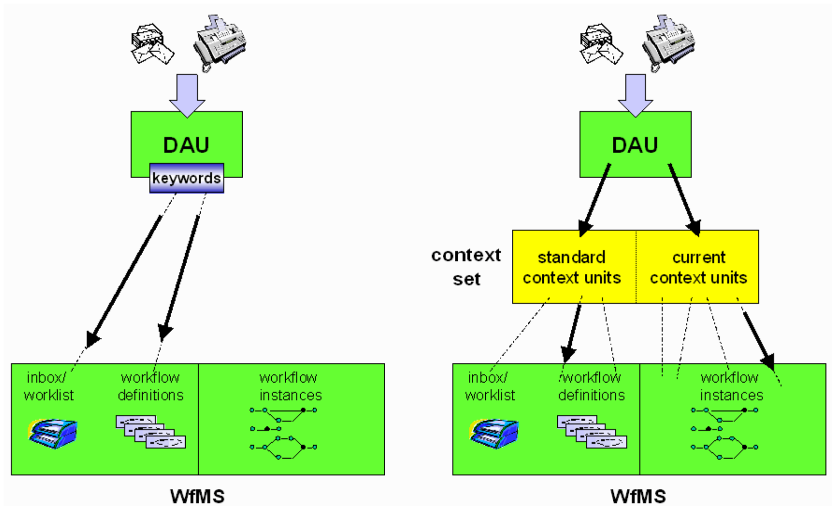


Fig. 1. State-of-the-art procedure in the analysis of printed business letters (*left-hand side*) in comparison to the VOPR approach (*right-hand side*)

German business letters. The Omega concept has been the basis for the development of the SmartFix product [8] which today serves several markets, e.g., insurance accounts.

In addition, a system described by Bläsius et al. [9] has been commercialized within the WANDO product [10] for the analysis of invoices. From a scientific point of view, it differs from the FRESCO approach mainly by the usage of the Dempster-Shafer approach instead of fuzzy sets for the propagation of uncertainty values.

Other research which might be listed within the context of knowledge-based document analyis has been done by Stephen Lam [11] who describes a frame-based approach especially for logical labeling. Further representation formalisms for generic document analysis purposes are based on predicate logic [12] and production rules [13].

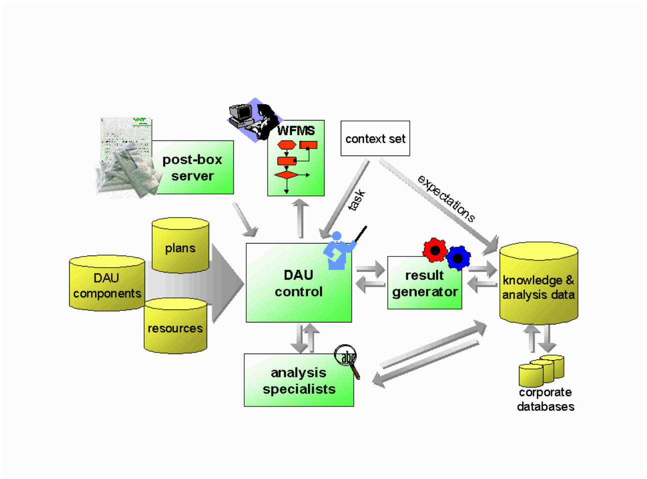


Fig. 2. System architecture of the VOPR system

3 System architecture

The full-fledged integration of a DAU system into a WfMS-based business environment is displayed in Fig. 2. Data flow starts, on the one hand, with a clerk's input to the WfMS (for the derivation of the application context and the analysis task) and, on the other hand, with new incoming documents (for the true analysis). New incoming documents already scanned and in an electronic representation are directly transferred to the DAU system by a post-box server.

The DAU system mainly contains components for a typical knowledge-based DAU: the central part is the DAU control which communicates with the post-box server 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 analysis tasks at hand. The analysis specialists use the knowledge and analysis data within the document knowledge and incorporated databases (e.g., postal addresses of suppliers) 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. Afterwards, the analysis control returns its results to the waiting workflow instance which presents them to the user. He or she corrects wrong DAU results which may later on lead to new document knowledge items derived by different learning components.

In order to enable the system to cope with context information, several components have been extended with appropriate interfaces. First, a WfMS is introduced and extended with a context set in order to deliver context information to the document knowledge by means of expectations. In addition, the DAU components are able to use these expectations for fulfilling their tasks more efficiently and more accurately (e.g., by search space reduction). Last, but not least, declarative analysis tasks are introduced to cope with the requirements of a dynamic environment.

The WfMS supplies the DAU system with context from workflow instances. This kind of corporate knowledge is annotated by entries of a separate corporate database. 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 by the analysis control according to the current task.

Within the following sections, we explain this procedure in more detail.

4 Retrieving context from workflows

The retrieval of context information from workflows for DAU purposes demands two prerequisites. First, an integration of the VOPR system into WfMS has to be established. Moreover, the necessary information pieces have to be modeled. These topics are the subject of *WfMS integration* and *Information representation* which follow. Given these prerequisites, the retrieval of context information divides into two major steps, namely *context collection within workflows* and *context transformation into context units* – a representation suitable for the DAU's knowledge base. The whole process is shown in Fig. 3 and detailed in the second part of this section. A specific case of context units – the so-called *standard context units* – finishes this section.

WfMS integration

Since today's (commercial) WfMS do not have modeling concepts for workflow context [14], there is no possibility to directly access context as a whole at runtime. However, there are several sources, e.g., control flow, data flow, application data, or a workflow's history [4], which allow the collection of context information for further processing.

Because the VOPR system is supposed to implement a generic integration into the WfMS independent of any proprietary extension to the workflow model, we introduce a database which we refer to as the *context pool.* Therein, all valuable context information is collected throughout the workflow execution. Technically, we model some special activities in the workflow definition which store involved information in the context pool, thus guaranteeing the availability of context when needed (for more details on the workflow integration concept please refer to [15]).

Information representation

In order to represent the documents produced within the workflow according to our business letter ontology, we need a representation for documents in the workflow. Furthermore, we need a means to model the domain ontology and to describe a document's contents with this ontology.

To this end, we use the World Wide Web Consortium's (W3C) standard XML (eXtensible Markup Language). XML is a markup language for documents, such as business letters, containing structured information. It provides means to define domain-specific, semantic, structured markup languages [16]. One such application of XML is RDF (resource description framework, [17]) which provides the means to describe metadata about (web) resources, and thus about our XML documents. RDF allows for the definition of domain-specific vocabularies in so-called *RDF schemas.* These schemas can be used as ontologies for XML documents. Therefore, the business letter domain is modeled as an RDF schema and is used to semantically enrich XML documents used in the workflow instances by relating the documents' data to this ontology.

For instance, Fig. 4 shows an excerpt from an order related to our business letter RDF schema (as can be seen within the first tag `rdf:Description`). The tag `Warenbezeichnung` represents a particular good which consists of several attributes. The one shown is the article name (`Artikelname`) with type `string` and value `CP 1.0-63-05L`.

Due to the possibility of using namespaces in XML, several different vocabularies can be incorporated within a single document and used throughout the workflow. This provides a powerful means for semantics in WfMS and for intelligent assistants as proposed in [18].

Given this, we are able to identify relevant information within the workflow and store it in the context pool. Thus, the information stored within the context pool is semantically described in terms of a domain ontology.

The following paragraphs will detail the aforementioned process of context generation.

Context collection within workflows

In case a situation 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. For instance, this is accomplished by the first two steps in Fig. 5 showing an excerpt of an actual workflow definition used for the VOPR system integration into the WfMS Staffware 2000 [19].

Therefore, the workflow stores any information about this event into the context pool such as the document itself. Afterwards, it invokes an inference engine and hands over some administrative data, such as the information need and the context pool reference of the causing event. Then the workflow waits until its expectation is satisfied, in our example, until the event INVOARVD is triggered.

Context transformation into a context unit

The concluding inference step uses the context pool as fact base for the generation of a context unit. The inference engine[2] converts the raw context information into a context unit by using transformation rules. These rules provide a mapping between the workflow's domain ontologies and a description of the expected document

[2] Actually, we use JESS (Java Export System Shell), see [21]

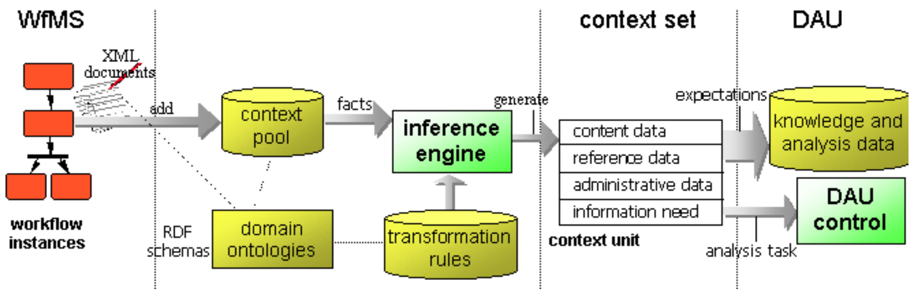


Fig. 3. Generation of context information from WfMS

```
<Warenbezeichnung>
<rdf:Description xmlns="http://www.dfki.de/NH/Geschaeftsbrief/Leistungsangaben/Warenangaben/
Warenbezeichnung.slots#">
     <Artikelname>
       <rdf:Description rdf:value="CP1.0-63-05L">
         <rdf:type rdf:resource="http://www.dfki.de/NH/Types#String"/>
       </rdf:Description>
     </Artikelname>
    ...
</rdf:Description>
</Warenbezeichnung>
```

Fig. 4. Excerpt from an XML/RDF order

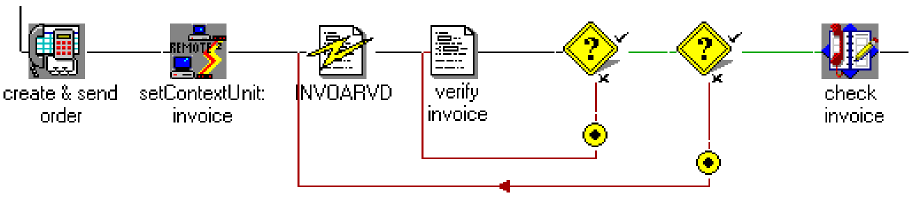


Fig. 5. Excerpt from a workflow definition with VOPR system integration

in terms of the DAU domain ontology (represented by the DAU's knowledge base). The exemplary rules given in Fig. 6 show the ontology mapping from a fact with an id given by our business letter RDF schema to a path for the document knowledge (the path notation is `<concept>(.<part>)*.message.<slot>`).

For instance, the first rule, named `Warenbezeichnung_Artikelname`, fires if a fact in the context pool has the given id. The rule header then binds type, value, and package to the respective variables where *package* denotes a number which indicates the grouping within a set-typed slot. The rule body performs several actions including the arrangement of the new concept path according to the DAU's ontology, the conversion of the value from the RDF schema to a value of the DAU's ontology, and finally, the insertion of the construct into the expectation contents of the current context unit.

To derive content and meaning of the expected document, some rules must also be domain-specific, e.g., the second exemplary rule stating that the recipient (`Empfänger`) of the outgoing document is to be the sender (`Absender`) of the document expected, or the last rule assembling information from the outgoing document to describe potential references within the answer, e.g., the date when a document has been written (`Schreibdatum`).

This information is stored within the content and reference data part of the context unit. Other rules generate administrative data or integrate the information need of the workflow into a context unit by stating the analysis task. The resulting context unit provides all necessary information in terms of the DAU's knowledge representation (to be detailed in Sect. 5). Figure 7 presents an exemplary context unit with paths in the German notation of the DAU's knowledge representation. For example, the path

`Geschäftsbrief.Leistungsangaben.Warenangaben.`
`Warenbezeichnung.message.Artikelname`

translates to

`businessLetter.service.goods.goodDescription.`
`message.articleName`

First, a context unit describes content and meaning of an expected document and its relationship to the business process which it refers to by stating all facts known in advance about a document. This may be the document's message type (1) (invoice = `Rechnung`) or the expected product list (2), because these products have been ordered in the corresponding business process. Furthermore, references to preceding documents are included, such as information from the corresponding offer (3).

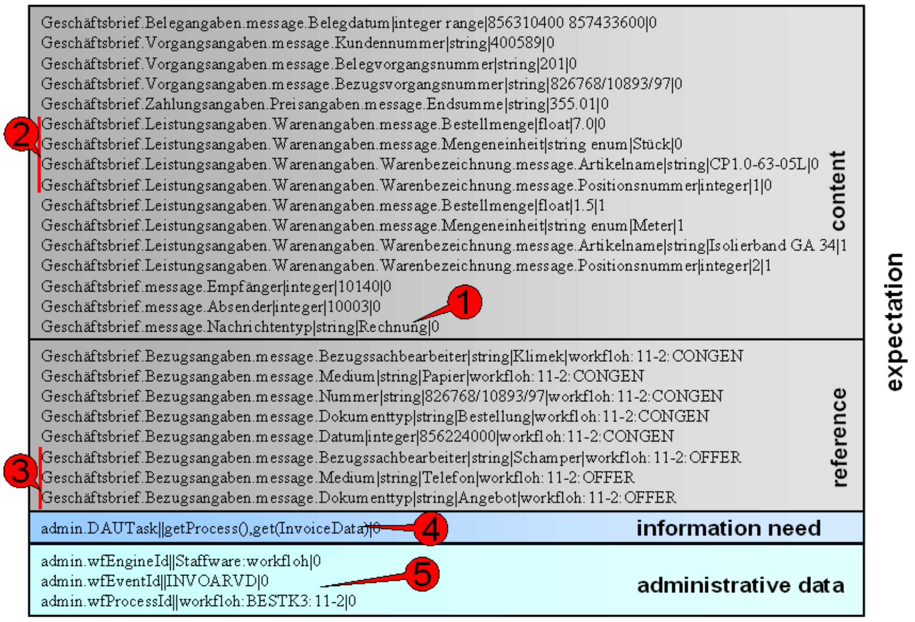
Moreover, a context unit expresses the information need of a business process. This is achieved by including analysis tasks which describe the required information (4). We distinguish two kinds of tasks: the first one is called *process identification* (`getProcess()`) and is given inherently in the 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* (`get(...)`) which requests specific information from the document necessary for further processing in the workflow. Information items required are either specified by a path notation revealing which information items have to be extracted or (as seen in the example) it is defined as a shortcut by macros which stand for a number of such paths.

```
(defrule  Warenbezeichnung_Artikelname
  (RDFFact    (id    "http://www.dfki.de/NH/Geschaeftsbrief/Leistungsangaben/Warenangaben/Warenbezeichnung.slots#Artikelname")
     (type ?t) (value ?v) (package ?p)
  =>
  (bind ?path (str-cat ?*concept* ".Leistungsangaben.Warenangaben.Warenbezeichnung.message.Artikelname"))
  (addEntry ?path string (convertValue ?t ?v) ?p ?*expectationContent*))

(defrule  Absender
  (unique (RDFFact (id "http://www.dfki.de/NH/Geschaeftsbrief.slots#Empfaenger") (type ?t) (value ?v) (package ?p)))
  =>
  (bind ?path (str-cat ?*concept* ".message.Absender"))
  (addEntry ?path integer (convertValue ?t ?v) ?p ?*expectationContent*))

(defrule Bezugsangaben_Belegdatum
  (RDFFact (id "http://www.dfki.de/NH/Geschaeftsbrief/Belegangaben.slots#Schreibdatum") (type ?t) (value ?v) (package ?p))
  =>
  (bind ?path (str-cat ?*concept* ".Bezugsangaben.message.Datum"))
  (addEntry ?path integer (convertValue ?t ?v) ?*cpBasedUponID* ?*expectationReference*))
```

Fig. 6. Simple exemplary rules



```
Geschäftsbrief.Belegangaben.message.Belegdatum|integer range|856310400 857433600|0
Geschäftsbrief.Vorgangsangaben.message.Kundennummer|string|400589|0
Geschäftsbrief.Vorgangsangaben.message.Belegvorgangsnummer|string|201|0
Geschäftsbrief.Vorgangsangaben.message.Bezugsvorgangsnummer|string|826768/10893/97|0
Geschäftsbrief.Zahlungsangaben.Preisangaben.message.Endsumme|string|355.01|0
Geschäftsbrief.Leistungsangaben.Warenangaben.message.Bestellmenge|float|7.0|0
Geschäftsbrief.Leistungsangaben.Warenangaben.message.Mengeneinheit|string enum|Stück|0
Geschäftsbrief.Leistungsangaben.Warenangaben.Warenbezeichnung.message.Artikelname|string|CP1.0-63-05L|0
Geschäftsbrief.Leistungsangaben.Warenangaben.Warenbezeichnung.message.Positionsnummer|integer|1|0
Geschäftsbrief.Leistungsangaben.Warenangaben.message.Bestellmenge|float|1.5|1
Geschäftsbrief.Leistungsangaben.Warenangaben.message.Mengeneinheit|string enum|Meter|1
Geschäftsbrief.Leistungsangaben.Warenangaben.Warenbezeichnung.message.Artikelname|string|Isolierband GA 34|1
Geschäftsbrief.Leistungsangaben.Warenangaben.Warenbezeichnung.message.Positionsnummer|integer|2|1
Geschäftsbrief.message.Empfänger|integer|10140|0
Geschäftsbrief.message.Absender|integer|10003|0
Geschäftsbrief.message.Nachrichtentyp|string|Rechnung|0

Geschäftsbrief.Bezugsangaben.message.Bezugssachbearbeiter|string|Klimek|workfloh: 11-2: CONGEN
Geschäftsbrief.Bezugsangaben.message.Medium|string|Papier|workfloh: 11-2: CONGEN
Geschäftsbrief.Bezugsangaben.message.Nummer|string|826768/10893/97|workfloh: 11-2:CONGEN
Geschäftsbrief.Bezugsangaben.message.Dokumenttyp|string|Bestellung|workfloh:11-2:CONGEN
Geschäftsbrief.Bezugsangaben.message.Datum|integer|856224000|workfloh:11-2:CONGEN
Geschäftsbrief.Bezugsangaben.message.Bezugssachbearbeiter|string|Schamper|workfloh: 11-2: OFFER
Geschäftsbrief.Bezugsangaben.message.Medium|string|Telefon|workfloh: 11-2: OFFER
Geschäftsbrief.Bezugsangaben.message.Dokumenttyp|string|Angebot|workfloh: 11-2: OFFER

admin.DAUTask||getProcess(),get(InvoiceData)|0

admin.wfEngineId||Staffware:workfloh|0
admin.wfEventId||INVOARVD|0
admin.wfProcessId||workfloh:BESTK3: 11-2|0
```

Fig. 7. An exemplary context unit

Lastly, the context unit lists administrative data to accomplish the integration into WfMS such as the identification of the workflow instance and the event to trigger if the document arrives (5), e.g., in Fig. 5 this would be triggering the event `INVOARVD`.

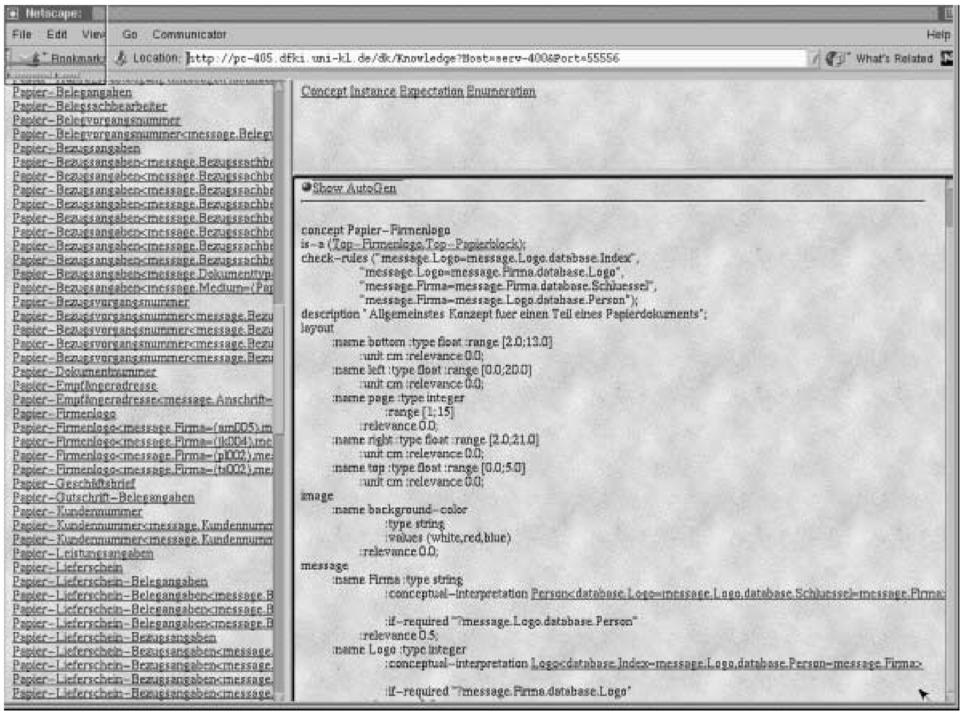
The context unit generated is stored within the context set. In addition, content and reference data are inserted as an expectation into the DAU's knowledge representation called *document knowledge*. Representation and usage of an expectation in the document knowledge is explained in Sect. 5.

Standard context units

Besides using dynamic context information from business processes, we also model more static context information by means of *standard context units*. These units serve 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 in two cases. If there is no match between an incoming document and context units from the WfMS, a standard context unit is the default match (e.g., for documents of type 'invoice'). Such a unit defines the default handling as, for instance, the instantiation of a workflow definition which will route an unassigned invoice to the accounting department for manual assignment. Second, standard context units catch documents referring to an existing process which are not expected such as reminders. Here, a specific analysis task has to determine the correct process in case that no explicit context unit exists for this document. Given this default handling for incoming documents, the VOPR system realizes the state-of-the-art procedure for mail processing (see Fig. 1).

5 Representing document knowledge

There is one central repository which stores generic document properties and analysis results called *document*


Netscape:
File Edit View Go Communicator Help
Bookmarks Location: http://pc-405.dfki.uni-kl.de/dk/Knowledge?Host=serv-400&Port=55556 What's Related
Concept Instance Expectation Enumeration
Show AutoGen

```
Papier-Belegangaben
Papier-Belegsachbearbeiter
Papier-Belegvorgangsnummer
Papier-Belegvorgangsnummer<message.Beleg
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=(sm005),m
Papier-Firmenlogo<message.Firma=(lk004),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. 8. Screenshot of the document knowledge browser for the concept company logo

knowledge. Besides storing typical document knowledge and data, some powerful mechanisms have been included which allow the storage of context information within the same formalism. This enables a direct context access for all analysis components and a semantic exploitation without doing any format adaptations. The remainder of this section will detail this.

Our knowledge 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 certain document properties by facets and facet values. Typical contents of document models are well-known in the document analysis community [21, 22]. Within our model, 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.

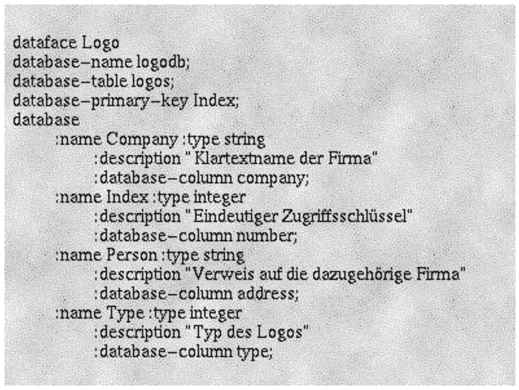
There are also different types of facets which may be used for slots, namely for representing uncertainty (`:certainty`), values and value types (`:value`, `:range`, `:unit`, …), related thesaurus information (`:thesaurus-name`, `:thesaurus-value`, …), frequencies, and relevances. Besides this, we also 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 importantly – 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*. One example for a company logo is given in Fig. 8. This figure shows a screenshot of our document 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 kind chosen which are currently defined (e.g., the first one `Papier-Belegangaben` means paper-record data). Last, but not least, the lower right area contains the definition of the frame chosen (`Papier-Firmen-logo` means a company logo on a paper document). The check-rules defined in this example state that the company the logo belongs to must be consistent with the sender of the document.

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*. Figure 9 shows the dataface for a Logo as another screenshot of the lower right area of our browser. This dataface relates slots within the document knowledge directly to columns of a database table. This allows transparency when accessing databases via the document knowledge because all database operations are hidden within the functionality of the document knowledge where they are implemented only once.

The second kind of context information relevant for DAU has already been mentioned: context in form of possible contents of expected documents. Looking at this 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 specializations of concepts already defined (e.g., an invoice for a printer of the com-



```
dataface Logo
database-name logodb;
database-table logos;
database-primary-key Index;
database
        :name Company :type string
                :description "Klartextname der Firma"
                :database-column company;
        :name Index :type integer
                :description "Eindeutiger Zugriffsschlüssel"
                :database-column number;
        :name Person :type string
                :description "Verweis auf die dazugehörige Firma"
                :database-column address;
        :name Type :type integer
                :description "Typ des Logos"
                :database-column type;
```

Fig. 9. Screenshot of the document knowledge browser for the dataface logo

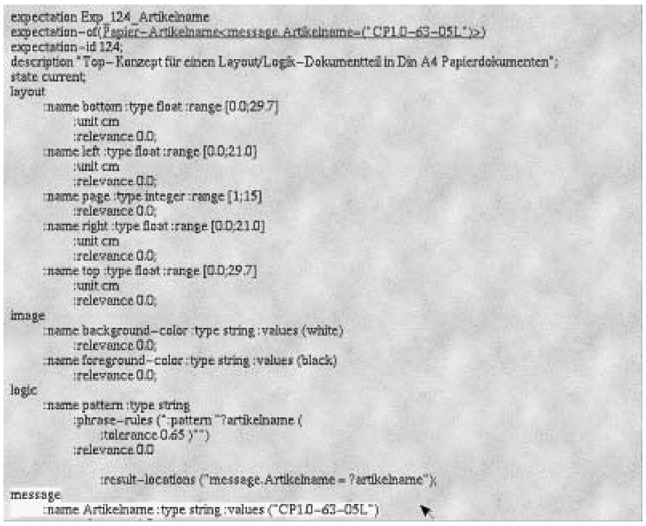
pany HewlettPackard is a specialisation of an invoice of the company HewlettPackard). An example is given in Fig. 10. Here, we have an expectation for a document part dealing with the name of an article ordered. This frame type is called *expectation*. The only entry in which this expectation differs 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 that 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. 11. 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.

Up to now, we have presented a rough overview on the functionalities of our document knowledge. The following sections will deal with how this representation is used during analysis.

6 Analysis control

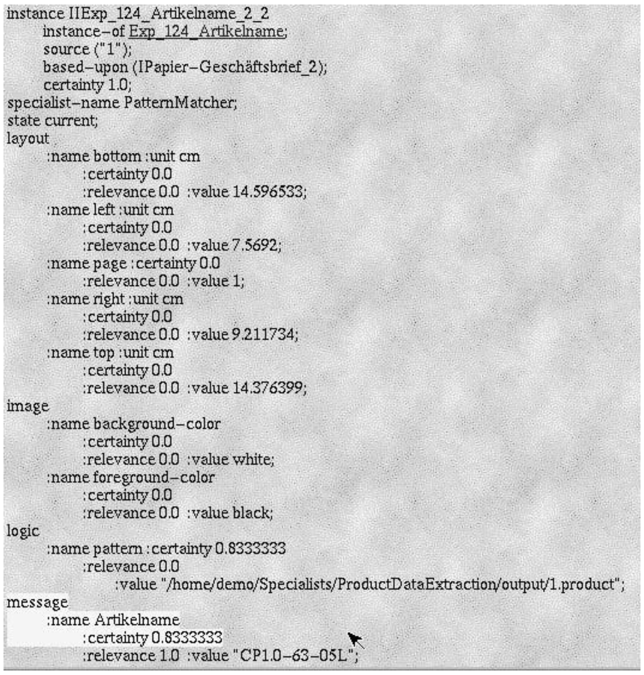
The DAU system is triggered in two different situations: in the first situation, the post-box server triggers DAU control with a new incoming document. In such a case, a process assignment has to be accomplished. The DAU control just retrieves the corresponding plan which denotes a sequence of DAU specialists along with pre- and postconditions and alternative paths. Figure 12 shows a simple analysis plan for the extraction of typical invoice data. 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 may extract which kind of information in a declarative way by using very 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. An example is given in Fig. 13 where the left-hand side shows the program path and the program parameters while the right-hand side shows the naming conventions for the document knowledge, the information items to be analyzed (concept-parts) and some component-specific declarative 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. 10. Screenshot of the document knowledge browser for an expectation of an article name concept space



```
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. 11. Screenshot of the document knowledge browser for an instance of an article name

Using resources and components, specialists are invoked and controlled according to plans. Of course, specialists which have already been executed for the document under consideration are filtered out since their results are already available. Having executed the analysis plan, the DAU control transfers the matching expectation id to the workflow management system. Hence, the document is assigned to this expectation and the corresponding context unit. If requested, additional extracted

```
plan-1:Plan( name="ExtractAdditionalInvoiceData", next={ start-1 })

start-1:Start( next={ specialist-1 })
ende-1:Ende( prev={ decider-1 })
ende-2:Ende( prev={ specialist-5 })

specialist-1:Specialist( name="ProductDataExtraction", next={ specialist-2 }, prev={ start-1 },
    infoitem={ Papier-Geschäftsbrief.Zahlungsangaben.Preisangaben })
specialist-2:Specialist( name="SenderExtraction", next={ specialist-3 }, prev={ specialist-1 })
specialist-3:Specialist( name="SenderEvaluation", next={ entscheider-1 }, prev={ specialist-2 })
specialist-4:Specialist( name="ShortSenderExtraction", next={ specialist-6 }, prev={ entscheider-1 })
specialist-5:Specialist( name="ResultGeneration", next={ ende-1 }, prev={ entscheider-1 })
specialist-6:Specialist( name="ResultGeneration", next={ ende-2 }, prev={ specialist-4 })

decider-1:Decider( if="Sender rating satisfying?", then={ specialist-5 }, else={ specialist-4 },
                   next={ specialist-5, specialist-4 })
```

Fig. 12. A plan for the extraction of invoice data

```
[ProductDataExtraction]
program = "/project/vo/bin/pmknow"    DPI-VALUE = 300
slot = "-s"                           LAYOUT-CREATOR = "vote"
pfad = "-p"                           WRITE-ONLY-VARIABLES = false
documentId = "-d"                     TRAPEZOID-WEIGHT = 0.8
expectation = "-e"                    SPECIALIST-NAME = "PatternMatcher"
accuracy = "-a"                       EXP-STRATEGY = "normal"
fast = "-f"                           PATTERN-SLOTGROUP = "logic"
section = "-n"                        PATTERN-SLOT = "pattern"
monitorLevel = "-l"                   PATTERN-FACET = "phrase-rules"
monitorChannel = "-c"                 RANGE-FACET = "range"
feedbackMessage = "-m"                PAGE-SLOT = "layout.page"
feedbackInstances = "-i"              LEFT-SLOT = "layout.left"
feedbackChannel = "-k"                OCR-SPECIALIST-NAME = "WORDCLASSIFIER1"
resourceFile = "-r"                   CONCEPT-PARTS = ( "Papier-Geschäftsbrief.Zahlungsangaben.Preisangaben"
orbHost = "-x"                        "Papier-Geschäftsbrief.Leistungsangaben.Warenangaben.
orbPort = "-y"                        Warenbezeichnung.Artikelname" )
remote = "serv-401"
...
```

Fig. 13. Partial resource entry for the extraction of product data

information is retrieved from the document knowledge (by making usage of its 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 the process. Such a new information extraction task is also specified by macros (e.g., the plan name in Fig. 12 is such a macro name). 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. As an example, see again the example plan where the specialist for product data extraction shall retrieve the prices of products (`Preisangaben`). The corresponding resource entry in Fig. 13 shows that this specialist would also be able to extract article names, but in the case of the example plan, this extraction is not carried out.

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. Figure 14 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. 15).

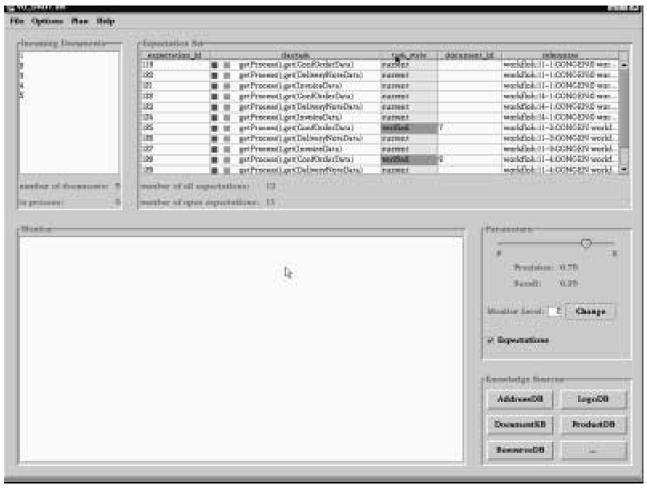
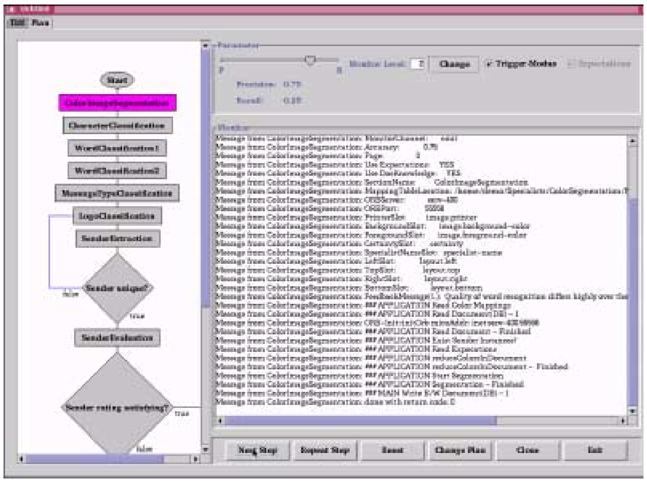


Fig. 14. Screenshot of the analysis control post

7 DAU components and DAU specialists

All DAU components which have been integrated into the VOPR system are displayed in Fig. 16. Those components which heavily rely on declarative document knowledge may be transformed into different domain- and task-specific specialists. For such components, the re-



```
Monitor Level: 7   [✓ Trigger-Modus]   [ ] Expectations
Precision: 0.75
Recall: 0.25
```

```
Message from ColorImageSegmentation: MonitorChannel: [illegible]
Message from ColorImageSegmentation: Accuracy: 0.75
Message from ColorImageSegmentation: Page: 3
Message from ColorImageSegmentation: Use Expectations: YES
Message from ColorImageSegmentation: Use DocKnowledge: YES
Message from ColorImageSegmentation: ServiceName: ColorImageSegmentation
Message from ColorImageSegmentation: MappingTableLocation: /home/[illegible]/Specialists/ColorSegmentation/[illegible]
Message from ColorImageSegmentation: OCRServer: serv-[illegible]
Message from ColorImageSegmentation: OCRPort: 55058
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[...]: 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: [illegible]
Message from ColorImageSegmentation: ### APPLICATION Read Document - Finished
Message from ColorImageSegmentation: ### APPLICATION Exist Sender Instance?
Message from ColorImageSegmentation: ### APPLICATION Read Expectations
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. 15. Screenshot of a plan execution

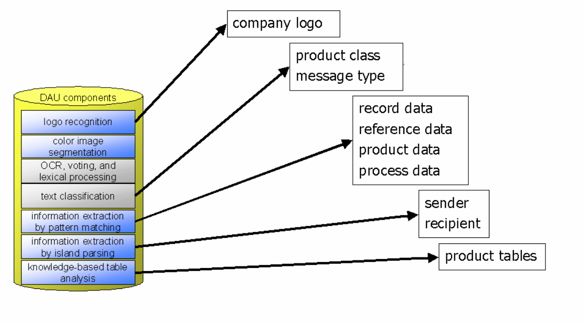


Fig. 16. DAU components (left-hand side) and resulting specialists (right-hand side)

sulting specialists currently employed are shown on the right-hand side of Fig. 16. Now a short description of each component 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 scannerprofiles. 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 this, 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. Our logo database contains at the moment more than 200 different 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 output of three commercial OCR engines (M/Text, TextBridge, and EasyReader) is 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, the lexical processing component 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: this component allows an error-tolerant, but shallow information extraction. Regular expressions for syntactic text patterns are processed in two steps whereby confidence measures from lexical processing (bottom-up) and from document knowledge (top-down) are combined. The component uses different similarity measures for words based on morphology (word stems, part-of-speech), semantics (synonyms, hypernyms), geometry, and fonts. The pattern matcher generates results for layout and message slotgroups of the document knowledge.

Parser: this component 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: this component 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.

For more information on these components, see [23–26]. There is another component which is not a true analysis component but typically included at the end of a plan:

Result generator: the result generator combines instances produced by other specialists and stored in the document knowledge 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.

8 Analysis strategies

Those components which can be transformed into several specialists are applicable in situations either with or without expectations. The current situation is submitted to them when being invoked. When dealing with

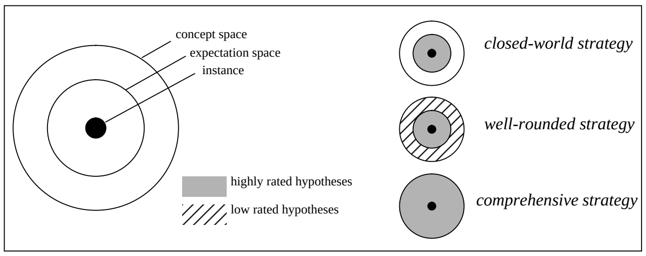


Fig. 17. Different analysis strategies and their general effect on results

expectations, several strategies can be used: the *closed world strategy* restricts the specialists' applications only to expectations, that means only results which instantiate expectations are produced. The second, *well-rounded strategy* allows results which are consistent with expectations while the last, *comprehensive strategy* allows the generation of both results based on expectations and results based on more general concepts at the same time.

Figure 17 explains these strategies with respect to the corresponding result space. The well-rounded strategy allows the different weighting of result hypotheses depending on whether they are based on expectations or not.

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 what unexpected documents may occur beneath those modeled in standard context units?).

We take our pattern matcher as an example for explaining the different strategies: this pattern matcher extracts information items based on text patterns which reveal textual, geometric, and semantic features of a textual expression. Imagine a simple pattern which determines how to extract a price ("total amount:" `?<NUMBER> currency`). This pattern extracts every number which is enclosed between the string "total amount:" and some currency (currency counts as a hypernym). Imagine also, that there is a price expectation which gives a concrete price (e.g., 105 USD). If running with expectations, the pattern matcher specializes the original pattern to "total amount:" ?105 "USD". When using the *closed-world strategy*, the pattern matcher only matches prices which amount to 105 USD. If no price expectation is given by the workflow, the pattern matcher matches no prices at all. However, the threshold for lexical verification of exactly this price might be set comparatively low to avoid rejects. Using the *well-rounded strategy*, the pattern matcher also matches the specialized pattern, however, if no price expectation is given, it matches the general price pattern with a lower a priori certainty. Using the *comprehensive strategy*, both the specialized and the general pattern are matched.

9 Experimental results

The generation of a proper testbed in order to prove the usefulness of expectations has been quite time-consuming. However, we could not simulate a realistic scenario with some hundreds of open expectations at a time. Efforts have to be undertaken 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 a time. For process identification, we identified 16 information items which may be at hand in expectations, e.g, sender, message type, process number, total amount, etc. 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 of the relevant information items. The results are shown in Table 1.

This table compares an expectation-guided analysis to a conventional run. Results are shown for some exemplary information items analyzed by different specialists and for the process identification. Not surprisingly, process identification with the usage of expectations was always correct. However, when neglecting expectations for analysis, the final analysis hypothesis led in four cases to a wrong process determination. As we see, the usage of expectations also leads to improvements in precision and in recall. Precision is improved since the number of possible solutions gets smaller, recall is improved since search can accept weaker solutions as correct ones. 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 (when 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 (and especially knowing, e.g., the correct number in advance) 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).

10 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 representation allowing to incorporate different knowledge sources such as corporate databases and

Table 1. Results for process identification and a few information items with and without expectations

	Logo recognition (precision)	Logo recognition (recall)	Message type (precision)	Message type (recall)	References (precision)	References (recall)	Accuracy of process identification
With expectations	100%	66.6%	90%	75%	94%	80%	100%
Without expectations	100%	58.3%	80%	66.6%	73.3%	55%	66.6%

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 learning capabilities by incorporating verification results (however, this was not subject of this paper). Additionally, the system incorporates some properties which allow for easy configuration:

- WfMS-vendor independent integration
- Free usage of domain ontologies within workflows
- Information need of workflows is freely definable
- Corporate dispatch strategies via context set
- Declarative and flexible formulation of tasks, results, and context
- Configurable specialists by component/resource combination
- DAU control based on plans
- Model-based document knowledge:
  - Domain independency due to models
  - Transparent integration of corporate databases
  - Inheritance and aggregation provide powerful, though modular concepts

The system accomplishes a tight integration of DAU into WfMS and therefore bridges the gap between paper-based parts of communication and automated business processes. The presented solution for context collection which considers current trends in e-commerce (XML) is efficient while being unintrusive. The only necessary efforts remain at buildtime by including some extra activities in the workflow definition for implementing the integration. The VOPR system is also suitable for requirements of business process reengineering due to its adaptability.

Our future work will further evaluate the impact of expectations on DAU results. On the one hand, we will assess the expenses of efforts for context collection against the revenue in analysis results. On the other hand, 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 (former, in part) 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. www.kofax.com
2. www.captivacorp.com
3. www.cardiff.com
4. S. Jablonsky, C. Bussler: Workflow Management. Modeling Concepts, Architecture and Implementation. International Thomson Computer, 1996
5. R. Bleisinger, M. Müller, P. Hartmann, T. Dörstling: Intelligente Eingangspostverarbeitung mit wissensbasierter Dokumentanalyse, Wirtschaftsinformatik Vol. 41, 1999, pp. 371–377, in German
6. T. Bayer: Understanding structured text documents by a model-based document analysis system. 2nd Int. Conf. on Document Analysis and Recognition, Tsukuba Science City, Japan, Oct. 1993, pp. 448–453
7. S. Baumann, M. Ben Hadj Ali, A. Dengel, T. Jäger, M. Malburg, A. Weigel, C. Wenzel: Message extraction from printed documents - a complete solution. 4th Int. Conf. on Document Analysis and Recognition (ICDAR 97), Ulm, Germany, August 18–20, 1997
8. Insiders Information Management GmbH: http://www.im-insiders.de
9. K.-H. Bläsius, B. Grawemeyer, I. John, N. Kuhn: Knowledge-based document analysis. 4th Int. Conf. on Document Analysis and Recognition, Ulm, Germany, August 1997, pp. 728–731
10. Asimus: http://www.asimus.de
11. S.W. Lam: An adaptive approach to document classification and understanding. 1st workshop on document analysis systems, Kaiserslautern, Germany, 1994, pp. 231–251
12. F. Esposito, D. Malerba, G. Semeraro, C. D. Antifora, G. de Gennaro: Information capture and semantic indexing of digital libraries through machine learning techniques. 4th Int. Conf. on Document Analysis and Recognition, Ulm, Germany, August 1997, pp. 722–727
13. R. Ingold: A document description language to drive document analysis. 1st Int. Conf. on Document Analysis and Recognition, Saint-Malo, France, 2 Sept./Oct., 1991, pp. 294–301
14. 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, Calif., USA
15. H. Maus: Towards a functional integration of document analysis and understanding in workflow management systems. Proc. Workflow Management Conference '99: Workflow-Based Applications, Münster, Germany, Nov. 1999
16. E.R. Harold: XML Bible. IDG Books Worldwide, 1999
17. Resource Description Framework: http://www.w3c.org/rdf

18. A. Abecker, A. Bernardi, H. Maus, M. Sintek, C. Wenzel: Information supply for business processes: coupling workflow with document analysis and information extraction. J. Knowledge-based Syst., Special Issue on Artificial Intelligence in Knowledge Management, vol. 13, issue 5, 2000
19. Staffware plc: http://www.staffware.com
20. Java Expert System Shell (JESS): http://herzberg.ca.sandia.gov/jess/
21. G. Nagy: What does a machine need to know to read a document? Symposium on Document Analysis and Information Retrieval, Las Vegas, Nev., USA, 1992, pp. 1–10
22. S. Srihari: From pixels to paragraphs: the use of models in text recognition. 2nd Int. Conf. on Document Analysis and Recognition, Tsukuba Science City, Japan, Oct. 1993
23. T. Jäger, A. Pies, A: Weigel: Global versus local combination of OCR results. Proc. 5th JCIS, Atlantic City, N.J., Feb./Mar. 2000
24. C. Wenzel, M. Malburg: Lexicon-driven information extraction from a document analysis view. Int. Workshop on Lexicon Driven Information Extraction, Frascati, Italy, July 1997
25. M. Junker, R. Hoch: An experimental evaluation of OCR text representations for learning document classifiers. Int. J. Doc. Anal. Recognition, 1(2), 1998
26. C. Wenzel, W. Tersteegen: Precise Table Recognition by Making Use of Reference Tables. In: Seong-Whan Lee, Yasuaki Nakano (Eds.), Lecture Notes in Computer Science, vol. 1655. Springer, Berlin Heidelberg New York, 1999

Claudia Wenzel received a diploma in computer science from the University Kaiserslautern, Germany. Since 1994, she has been working at the German Research Centre for Artificial Intelligence (DFKI) in Kaiserslautern as a member of the document analysis and information management department. Her personal research interests deal with the textual exploitation of documents and especially techniques for text categorization, table analysis, and information extraction from paper documents.

Heiko Maus received a diploma in computer science from the University of Saarland, Saarbruecken, Germany. Since 1997, he has been working at the German Research Centre for Artificial Intelligence (DFKI) in Kaiserslautern as a member of the document analysis and information management department. His research interests include the combination of knowledge and workflow management with focus on the integration of knowledge intensive applications and context modeling.