GMD Report 48

GMD –
Forschungszentrum
Informationstechnik
GmbH

Bertin Klein, Andreas Abecker

Distributed Knowledge-Based Parsing for Document Analysis and Understanding

January 1999

GMD –
Forschungszentrum Informationstechnik GmbH
Schloß Birlinghoven
D-53754 Sankt Augustin
Germany
Telefon +49 -2241 -14 -0
Telefax +49 -2241 -14 -2618
http://www.gmd.de

Anschriften der Verfasser/Addresses of the authors:
Bertin Klein
Institut für Integrierte Publikations- und Informationssysteme
GMD – Forschungszentrum Informationstechnik GmbH
Dolivostraße 15
D-64293 Darmstadt
E-mail: Bertin.Klein@gmd.de

Andreas Abecker
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH
Postfach 2080
D-67608 Kaiserslautern
E-mail: Andreas.Abecker@dfki.de

ISSN 1435-2702

Abstract

Document Analysis and Understanding (DAU) is a complex AI application with high industrial impact. For the increasing demands upon the bandwidth and quality of the analysis it is crucial to enable different analysis modules to collaborate. For making collaboration possible, we first examine the question whether there exists a common ontological basis which can serve as a platform for communication of different DAU modules. Once communication is enabled, we investigate the second question, how DAU modules originally designed as stand-alone systems must be modified in order to benefit from collaboration with others.

Keywords: document analysis, document understanding, logical structure recognition, parser, module communication, module cooperation, ontologies

Zusammenfassung

Im Forschungsgebiet des *Document Analysis and Understanding* (DAU) beschäftigt man sich mit komplexen Applikationen aus dem Bereich der Künstlichen Intelligenz, die in vielen industriellen Anwendungen von Bedeutung sind. Eine Schlüsselrolle für die zunehmenden Anforderungen an die Bandbreite und die Qualität der Analyse kommt der Kombination verschiedener Module zur Kooperation zu. Mit dem Ziel effektive Kooperation von Modulen zu erreichen, wird zunächst untersucht, ob sich eine gemeinsame ontologische Basis bestimmen läßt, die als Kommunikationsplattform zwischen verschiedenen DAU Modulen dienen kann. Weiterhin wird untersucht, wie DAU Module, die bisher als stand-alone Systeme konzipiert und implementiert sind, modifiziert werden müssen, um von der Kooperation mit anderen Modulen profitieren zu können.

Schlagworte: Dokument Analyse, Dokument Verstehen, Erkennung logischer Struktur, Parser, Modul Kommunikation, Modul Kooperation, Ontologien

1 Introduction

Jochum [6] has proposed to "replace the notion of *information* with the notion of *text*" (original in German: *Schrift*), which others already rejected, like Umstätter [11]. Anyway, text is a very important and very popular carrier of information. This is of course the reason, why documents and libraries are such an important player in the emerging "distributed knowledge environments". The goal of *Document Analysis and Understanding* (DAU) is to extract from (paper or electronic) documents the information needed for subsequent processes and applications. In particular, this goal comprises to recognize the document's logical structure (title, author, abstract, introduction etc..) as a core activity and important prerequisite for subsequent information extraction steps (cf. Figure 1, slightly adapted from Dengel [3]). DAU plays an important, yet increasing role in tomorrow's information infrastructures both for translating paper documents into electronic form for further processing (Baumann et al [2]) and for automatic information extraction from Internet pages found by information gathering web-bots (Lesser et al [9]).

Today's DAU systems exhibit multi-module chain architectures with fixed sequences of specialized analysis steps (see Figure 1).

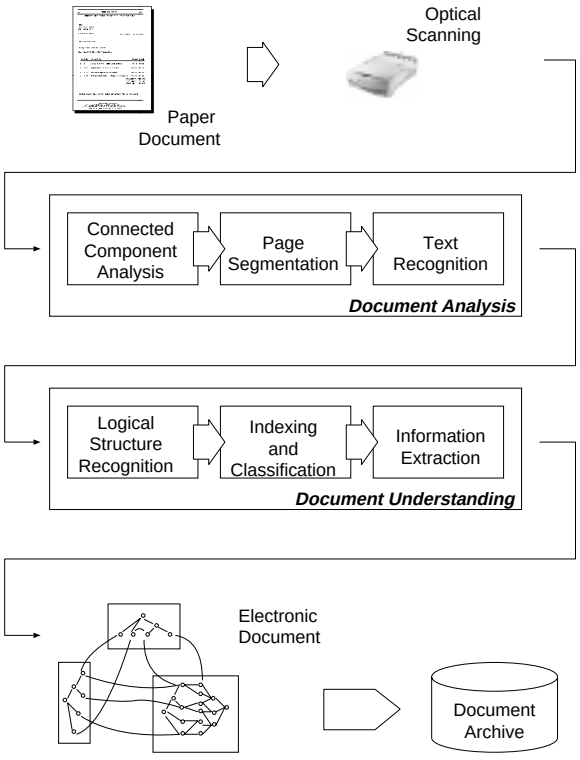

Figure 1: *Subtasks of Document Analysis and Understanding*

For Logical Structure Recognition (LSR) and markup, we developed the DREAM system, a state-transition machine (STM) parser [8]. Document structure is described by grammars that extend SGML with recognition rules. From these grammars, parsing automata are generated. These automata are used to partition a flat text document into its elements, to discard formatting information, and to insert SGML markup (cf. Figure 2). The chosen approach is not bound to a limited subset of the context-free grammars, such as LALR(1) (cf. Yacc) or LL(k) (cf. PCCTS),

and does not require a pre-processing of the input by a tokenizer, but operates directly on the document input.

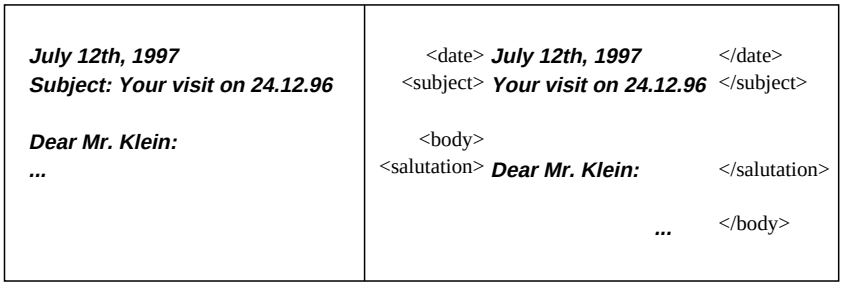

Figure 2: *DREAM Input and Output*

DAU applications prosper because their great potential in saving repetitive, error-prone, manual work. But DAU has the omnipresent problem of brittleness. Applications must be carefully configured and work only in very restricted problem domains. Towards the borders of the domain the analysis efficiency decreases vastly, or the analysis quality, or both. This ressembles the known problems of Artificial Intelligence. To identify and overcome the current barrier is the goal of this paper.

Great enhancements of both analysis efficiency and quality can be obtained, if the strict sequence of processing is broken up. Then such an LSR module would not be any more only an isolated step in the DAU process taking the input of the prior step and passing it as a whole to its successor, but could freely exchange information and intermediate results with other steps in the DAU process.

Consider the search for the date and the subject in a standard business letter. It is obviously opportune to search at specific spots instead of completely processing the whole document (transforming pixels to characters, interpreting the characters, and so forth). Thus, in this case, the low-level components should not be called in a hard-wired manner before the higher-level modules; instead the lower-level analysis should be controlled by a high-level module which has knowledge about spots of date and subject.

To give one more example, consider that, again in a letter, a fragment of a word is determined: "...street", by a high-level module. This information might be very valuable before the pixels around this word are processed, because a specific low-level OCR (Optical Character Recognition) tool could be applied which is specialized on numbers (of the house, as well as some zip code following to the right, or in the next line). This strategy might seem familiar, because experiments have shown, that our eyes while reading do so as well, "guess what comes next".

In this paper, we investigate cooperation between software modules in the DAU process. Cooperation can take place across different layers in Figure 1 as well as between different modules employed for one specific process step. Although our current implementation is focussed on Logical Structure Recognition, we are convinced that the ideas generalize to the complete DAU cycle.

The paper is organized as follows: Section 2 motivates the cooperation idea and explains the expectations. Then in Section 3 the theoretical prerequisites for cooperation are discussed. Section 4, by presenting the "Document Analysis Core Ontology", shows that the prerequisites for cooperation can be met in Document Analysis. Section 5 reviews this intermediate result and identifies how to use it and transfer it to practical systems, for which two sub-problems are distinguished: First in Section 6 it is looked at existing modules and how they must be adapted

for cooperation, whereas in Section 7 it is looked at new functionality which must connect the existing modules. Section 8 gives a summary and sketches future efforts.

2 Towards Combinable LSR Modules

DAU modules are brittle. The ambiguity of document structures and the need for fallback rules to facilitate error tolerant parsing make DREAM parsing automata highly complex. Sophisticated control strategies must be introduced in order to prune exponential search spaces.

Unfortunately, there is no "single, best strategy and document-structure grammar", but the performance of a <strategy, knowledge-base (grammar)> pair depends on the application domain, and on the specific document instance, as well. On the other hand, as we pointed out above, at a given point of the application, help may come from several possible supporting knowledge sources:

- Other LSR modules containing different document knowledge and pursuing other search strategies can give valuable hints complementary to the actual analysis results.
- Additional external knowledge sources can support the parsing process; when analysing business letters, e.g., a customer database can clarify whether the suspected "sender" entry in an invoice letter is a known customer.
- Other stages of the overall DAU process hold other points of view on the document, thus exploiting different facets of document knowledge.

Such a multitude of possible information suppliers suggests to further develop DAU technology towards collaborating modules which consult each other in difficult analysis situations thus taking advantage from the particular strenghts of several modules. Basically, the knowledge encoded into the grammar rules of two different LSR systems can in principle cooperate if the intersection of their conceptualizations (and thus, also their manifestations in terms of ontologies) is non-empty (cp. [13]).

In order to answer the question whether contemporary systems have such a non-empty intersection, we did a broad analysis of the existing systems for Logical Structure Recognition, essentially relying on two methods.

First, we had intense discussions with members of several German research groups active in the area; the goal was to reveal basic principles and concepts shared by all approaches which could be identified as starting points for cooperation. By discussing how in detail the systems handle and act with their concepts, we were able to agree on basic concepts which underlie all the different systems. In simple words, all the systems use one kind of concept to denote and trigger certain search actions in the analysis data, and another kind of concept to denote and derive the sequences for the search actions.

Second, we also did an extensive literature study to give the results an international dimension. The systems considered in detail are listed in Table 1.[1] The strategy was the same, only that we had to rely on our interpretation of how the systems work, which is probably not as authentical as face to face discussions. However, we found again the same concepts.

Thus the result of these studies was that there is in fact some shared point of view which is explained in the following sections.

[1] Systems not labeled with a proper name are denoted with their designer's name in brackets.

Name	Institution Reference
GRAPHEIN	CRIN/INRIA Lorraine, France Chenevoy & A. Belaid, ICDAR '91
Π_{ODA}	DFKI Kaiserslautern, Germany Bleisinger, Hoch & Dengel, DFKI Document, 1991
DSL	IBM Almaden Research Center, San Jose, USA Lorie, DAS '94
Page Grammars	Hitachi Dublin Institute, Dublin, Ireland Conway, ICDAR '93
(Kelly)	Department of Computer Science, Trinity College Dublin, Ireland Kelly & Abrahamson, ICDAR '91
(Spitz)	Daimler Benz Research, Palo Alto, California Spitz, ICDAR '91
(Hu)	University of Fribourg, Switzerland Hu & Ingold, Electronic Publishing 6(4), 1991
Fresco	Daimler Benz Research, Ulm, Germany Bayer, Bohnacker & Mogg-Schneider, DAS '94
(Kerpedjiev)	Institute of Mathematics, Sofia, Bulgaria Kerpedjiev, ICDAR '91
IDA	Siemens AG, Munich, Germany Kreich, ICDAR '93
DREAM	GMD-IPSI, Darmstadt, Germany Klein & Fankhauser, IEEE ADL '97

Table 1: *Examined Document Analysis Systems*

3 Ontological Prerequisites for Cooperation

Cooperation is beneficial, so that it is most often likely to outweigh the effort for its implementation. Many systems cannot cooperate very good. The remedy starts with understanding the prerequisites for cooperation.

"It is impossible to represent the world in its full richness of detail." (see [13]). Thus, for building knowledgeable systems attention has to be restricted to a certain number of concepts. Hence, "every knowledge base, knowledge-based system, or knowledge-level agent is committed to some conceptualization, explicitly or implicitly." (see [5]). If the conceptualization is explicit, it is called an *ontology.*[2] Two implications are important here:

1. An ontology exists for *each* of these systems.
2. An ontology is important to know because of the commitments it expresses.

This motivates our approach: From the examined systems we conclude their conceptualizations. The main interest is on the intersection of conceptualizations. The ontology describing the intersection is automatically part of any ontology of a DAU, or LSR system, respectively. This ontology reflects the similarities of the systems considered, and is the necessary basis for their cooperation. If two systems have fundamentally different ontologies, they certainly cannot cooperate. If cooperation is possible in principle, two further prerequisites are needed:

1. The systems must be able to communicate, and

[2]If this definition is not clear enough; we use *ontology* as: "any kind of collection of concept definitions". This can be glossaries, thesauri, formal knowledge bases etc.. This explicitly includes non-formal kinds of concept definitions.

2. it must be possible to combine the results of two systems.

While the former requires "a common ontology [, which] defines the vocabulary with which queries and assertions are exchanged among [systems]" (Gruber [5]), the latter requires even more. The two ontologies must expose a more elaborate kind of similarity, which exactly is one of the most important issues in this article.

Assume two systems, specialized on notice letters (of phone contracts of a specific company), which are able to cooperate in order to determine the letters' legal dates more safely. Obviously, this requires at least that both systems have, e.g., a concept of the date of notice letters, which Gruber [5] called part of the "exchange vocabulary". Now, this concept and the rest of the vocabulary are bound to the application. It is not known whether after a change of the application (e.g., to invoices of another company) cooperation is still possible. If we believe that cooperation is still possible (even though the mass of numbers in invoices might pose specific problems), then we assume a kind of similarity of how the systems work even on the new task, and we should be able to capture the kind of similarity in an ontology of shared concepts.

We first sketch those shared concepts before we discuss in more detail communication aspects and combination of results.

4 The Document Analysis Core Ontology (DACO)

DACO proves that DAU systems can cooperate (because this can be established on the basis of DACO).

Business letters contain instances of "address". If this simple fact is to be specified in a DAU system, suddenly "address" in the different formalisms takes on many different outward appearances, like a *frame, block, bounding box, composite-physical-type, symbol, nonterminal,* etc.! It seems that all these notions distract from the mere fact of interest.

The interesting fact at hand can be captured using DACO, our Document Analysis Core Ontology: DACO's concept of ELEMENT abstracts away from the syntactical details of the above notions. If we state that in business letters instances of the ELEMENT "address" are contained, this means that something exists which is called "address", and which can be reasoned about in its own right. We are not bothered with the question which requirements and boundaries the respective system imposes on the reasoning; i.e. we do not care about operating system, programming language, reasoning paradigm.

This argumentation is along the lines of the "knowledge level" which was introduced by Newell [10].

Because DACO is derived from the current systems' formalisms as their common denominator, it captures their similarities and is biased towards current DAU systems. One example is the ELEMENT: all systems' reasoning evolves around ELEMENTs like "address"; another example is the absence of a concept of time: documents typically do not change over time.

Entirely, DACO contains four basic concepts: ELEMENT, RELATION, ANNOTATION, and PROCEDURE. For their presentation we use the format introduced in the Enterprise project [12].

ELEMENT: an entity occuring in documents (electronic documents or paper documents).

Examples:

- a page is an ELEMENT

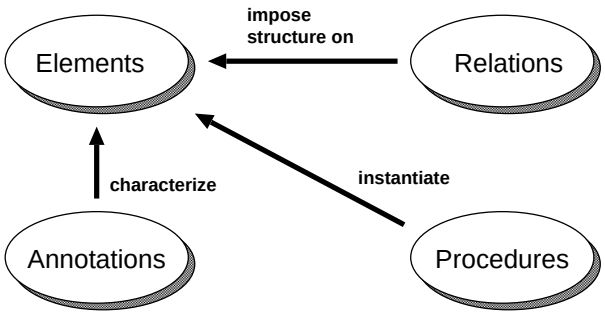

Figure 3: *The Basic Entities Constituting our Document Analysis Core Ontology DACO*

- a title is an ELEMENT

Notes:

1. A document is also an ELEMENT.
2. This concept abstracts from the two particular concepts of the *logical element* and the *layout element*.

RELATION: the way that two or more ELEMENTs can be associated with each other.

Examples:

- Part-of is a RELATION between two ELEMENTs to describe that one is a constituent of the other.
- Below and right-of are RELATIONs which can be defined by Allen's Relations applied to two axes of coordinates.

Notes:

1. A RELATION is not an ELEMENT that can participate in further RELATIONs.
2. The most important kinds of RELATIONs are the part-of and the is-a RELATION. For both of them it is known that they have different subcategories. Using two different subcategories can destroy the transitivity: e.g., an arm is part-of a person, which is part-of a group, but it is strange to conclude that the arm is part-of the group.

ANNOTATION: symbolic description of a property of an ELEMENT.

Examples:

- a symbolic content-description, like "graphics" is an ANNOTATION.
- a regular expression, like "Section [0-9.]+" is an ANNOTATION.

Notes:

1. The systems interpret ANNOTATIONs with their built-in functionalities thus mapping pieces of input to ELEMENTs. (If a system had no built-in functionality but a color sensor, ANNOTATIONs would be "red", "blue", etc.)
2. ANNOTATIONs actually used in the DAU systems are nothing but: *layout styles* (font-families, font-size, etc.) and *patterns* and/or *lexicons* (typical character sequences and words).

PROCEDURE: explicit statement how to reason in the process of analyzing an ELEMENT.

Examples:

- A system call like "./library/spellchecker" is a PROCEDURE.
- "Top-Down" is a PROCEDURE.

Notes:

1. Overriding default parameters "Top-Down" results in a modified procedure.
2. This concept is necessary to capture the systems DSL, Fresco, and DREAM which implement the first kind of procedure, and GRAPHEIN which switches between top-down and bottom-up parsing.

To sum up: all abstract or concrete document parts and properties used within a parsing run are represented as ELEMENTs. Essentially, document analysis knowledge is expressed by RELATIONs describing the numerous ambiguous possibilities how simple ELEMENTs can be aggregated into more complex ones, or vice versa, simple document features can be derived from more complex ones. A parsing run follows a search strategy determined by PROCEDUREs attached to processed ELEMENTs for building up a document parse tree representing the logical structure of the input document. At the leaf nodes of this parse tree there are terminal ELEMENTs testing the given document input data for certain properties with the help of ANNOTATIONs.

DACO has the characteristics of a[3]:

- **Meta Ontology** in the sense of Uschold & Grüninger because it comprises the main concepts to be used as the primitives for the definition of further concepts. [12].
- **Principled Core Ontology** in the sense of Valente & Breuker [13] because it is a "very general ontology of a certain application domain" and moreover it adheres to the four principles of parsimony, theoretical soundness, completeness, and coherence.
- **Domain Ontology** in the sense of van Heijst *et al.* [14] because it "expresses conceptualizations specific for the [document analysis] domain [...] [and] puts constraints on the structure and contents of DAU domain knowledge".

For this article the most important property of DACO is that it is the common denominator of current document analysis knowledge representations. All knowledge representations of systems adhere to conceptualizations, even if these are currently not described by ontologies. We have elaborated the common denominator of the conceptualizations and have described it with DACO. Thus, DACO will occur in any ontology which is set up for DAU knowledge representations. This shared part is the basis for system combination.

5 Requirements for Combinable DAU Modules

The DACO ensures that *communication* between several *human members* of the DAU research community is possible. This is already a valuable use of an ontology

[3] For further details on this classification, see [7].

according to Uschold and Gruninger [12]. The *interoperability* between *systems and modules*, and thus design of *reusable components* grounds on this communication basis too, but is by far not yet reached. The following ingredients are needed now:

1. A *metastrategy* to recognize situations where collaboration should be initiated.
2. A *mediation service* to find a module presumably able to help in the current situation.
3. A *communication service* to allow different modules to talk about inquiries and answers.
4. A *modified control strategy* to make beneficial use from answers of consulted modules.

These ingredients split into the *intra-module* and the *inter-module* aspects. The first and the last ingredient concern changes which must be done within the modules. The two inner issues concern aspects of module collaboration.

To enable reuse of problem-solving methods (PSMs), Fensel recently proposed the architecture underlying Figure 7 (cf. [4]) which illustrates our current cooperation scenario: a central DREAM system (DAU Module_1) starts parsing and runs until the metastrategy detects a cooperation need. This need causes a request to the mediation layer. The mediation service selects an appropriate supporting module (DAU Module_2) and establishes a connection between the two modules. The *adapter agent* of the mediation layer transforms different terminologies to enable knowledge and data exchange between different DAU modules. This transformation is enabled by the *ontology agent* holding the actual DAU application ontology together with mapping information for the several modules involved.

This cooperation scenario is based on inter-module communication about DACO ELEMENTs. However, contemporary, implemented DAU modules work on a too coarse-grained level for even considering single ELEMENTs. Thus, we have to investigate before how a given module must be prepared for cooperation (intra-module level).

6 Intra-Module Level: Metastrategy and Modified Control

The main steps to enable communication between modules were discussed in the previous sections. To turn theory to practice the current DAU modules have to be looked at now: They are designed for stand-alone use and they cannot be integrated into collaboration scenarios without considerable modifications.

The main point is illustrated in Figure 4, referring to DREAM. Today's DAU modules execute an analysis process in one coherent run which does not allow for intervention within the process. They perform a complex search, often including backtracking, based upon an agenda of current hypotheses. On the basis of the DACO, we propose to divide the reasoning into smaller entities: each node in the search tree represents an ELEMENT to be analysed and the hypotheses for interpreting it. Now, at each such point the reasoning process should be stopped and foreign modules should be consulted if this promises to improve the current information state. So, we introduced a *metastrategy* in our DREAM system which observes the size of the agenda and initiates external queries if the agenda is

- either too large for an ELEMENT, or
- empty (i.e., the parse is in a dead-end).

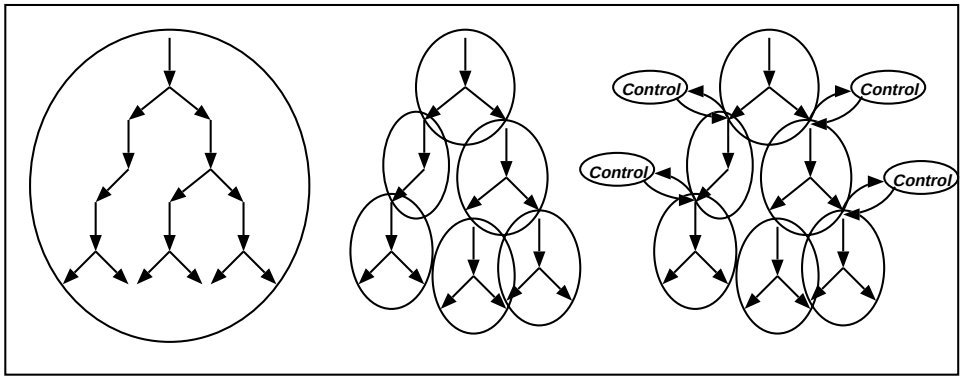

Figure 4: *Decomposition of the Transformation/Execution of a DREAM Syntax Tree And Insertion of Control Steps*

Extending this heuristic regarding the number of hypotheses, we currently consider also more sophisticated metastrategies. These can, e.g., be based upon ANNOTATIONs which allow the user to indicate that the modelled knowledge in a certain area is weak, such that support would be helpful.

Besides the described method for identifying possibilities for external queries, we need a *modified control* which is capable to integrate the incoming answers into the local reasoning process. In a full-fledged "multi-agent" DAU scenario, answers may be received from different systems with their respective strong and weak points, each of them claiming for a certain amount of uncertainty. So, adding and deleting hypotheses from the agenda must take into account this aggregated uncertainty.

We are currently investigating a simple qualitative specification for the degree of confidence an analysis module has in its results (so-called *belief tokens*, like "certain", "promising", "uncertain", etc.). The modified control must itself be based upon belief tokens and can thus easily incorporate such foreign, qualified results. This does not necessarily mean that each DAU module must be working along this idea. If a module acts only as a supporting agent which delivers hypotheses and answers, the specification can also be added by the mediation service; the mediation service must be able to estimate the quality of a module's results anyway in order to select a promising supporting module.

Introducing belief tokens also yields another basis for the metastrategy: the values allows to evaluate the current state of information which in turn allows to estimate the need for collaboration.

7 Inter-Module Level: Mediation and Communication Services

Up to now, we have modified our DREAM parser as described above such that it collaborates either with external databases or with another DREAM system (pursuing, e.g., another search strategy). In this case, inter-module communication is manageable because the systems do rely on the same ontological commitments, use the same knowledge representation, and employ the same data structures. Furthermore, it is easy enough to assess best partners for cooperation and hard-wire them in ANNOTATIONs and PROCEDUREs. This framework is ready for extension to sophisticated scenarios, in which systems can try to find needed information rather

autonomously. In the general case, these issues are a demanding task the solutions for which can build upon results from:

- multi-agent systems regarding the communication language,
- knowledge engineering, particularly reuse of problem-solving methods, regarding formats for competence descriptions of modules, and
- document analysis and understanding regarding the DAU application ontology as well as concrete competencies of implemented systems

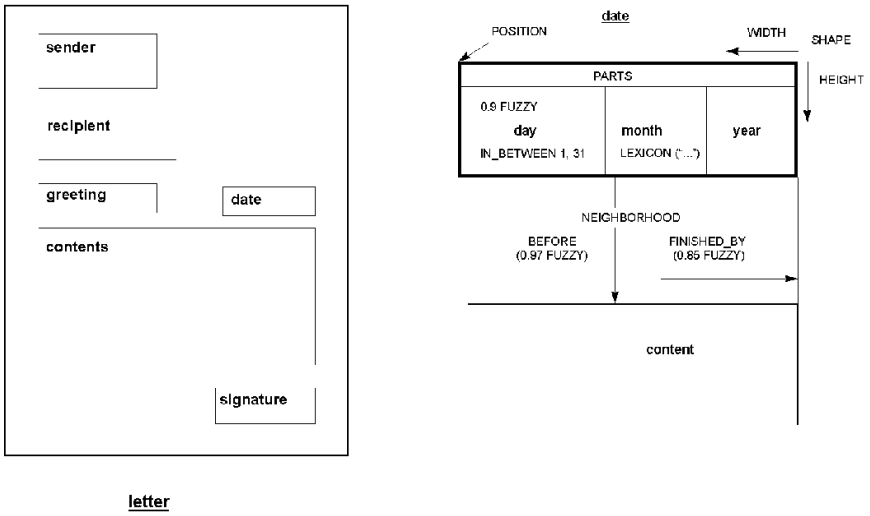

Figure 5: *Schema of a Business Letter and Exemplary Layout Knowledge*

Agent communication languages like KQML provide a *communication service*. The communication language must be able to express the speech acts needed for requesting and verifying hypotheses about ELEMENTs, the answers of both requests being equipped with belief tokens.

To transport the content of a request or an answer, input data structures (in the case of ELEMENTs which are direct parts of the input document) and knowledge-level module vocabulary (in the case of derived, abstract ELEMENTs contained in the document, like a date or an address field) must be translated between different modules. This is the job of an ontology server having at its disposal a DAU application ontology. In contrast to the DACO which seems relatively stable, this application ontology central to communication is still heavily under discussion.

In order to give an impression of what parts such an ontology must contain, consider Figure 5. On the left hand side, we have some ELEMENTs of a typical business letter. The right part of the figure demonstrates some layout-based document analysis knowledge, like, e.g., the fact that the date uses to be above the content part. Figure 6 sketches the ontological basis for expressing such knowledge: one part of the DAU application ontology concerns layout issues which in turn import some spatial notions for expressing spatial relationships between ELEMENTs.

The last requirement for establishing a connection between DAU modules is to identify the module(s) which will probably produce the most useful answer(s) to a given request. In our current implementation, this task is not critical because all external knowledge sources are either "specialists" (the appropriateness of which is

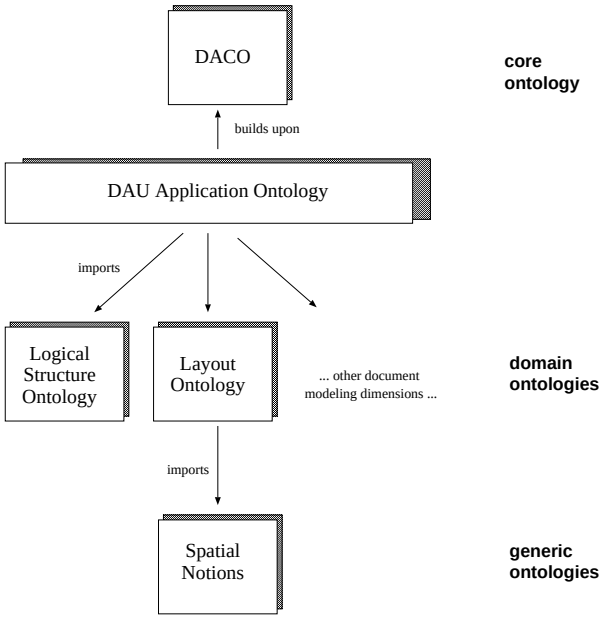

Figure 6: *Some Constituents of a DAU Application Ontology*

rather simple to assess) or another DREAM system. In the general case, with a number of available DAU systems coming from different institutions and focussing on different DAU aspects, the mediation service must know what system is competent for what question.

While the basic knowledge structures needed for providing such a competence assessment are investigated in the PSM community, concrete knowledge about strong and weak points of implemented DAU systems must be gathered within the DAU community.

Figure 7 summarizes our approach: a central, ontology-based mediation layer establishes connections and transforms input data and analysis vocabulary between cooperating modules. The modules are prepared for cooperation by an appropriately modified control together with a metastrategy starting requests and processing the answers.

8 Summary and Future Work

Logical Structure Recognition, and Document Analysis and Understanding in general, are complex AI applications with high industrial impact. In the area of Digital Libraries and Organizational Memories (which can be understood in many aspects as company-specific DLs, in a similar way as Intranets are related to the Internet) at least two purposes of LSR and DAU are of particular importance: building electronic repositories from large existing paper-based archives, and continuously filling and updating electronic archives with input from streams of text-based information items like, e.g., articles from news agencies sent by fax or e-mail.

We pointed out the need for collaboration of modules in order to improve results and to master complexity of analyses. We sketched the high potential of combining expertise and specific strengths of several cooperating tools. Apparently, our approach can be transferred to other, isomorphic problems typically attacked by

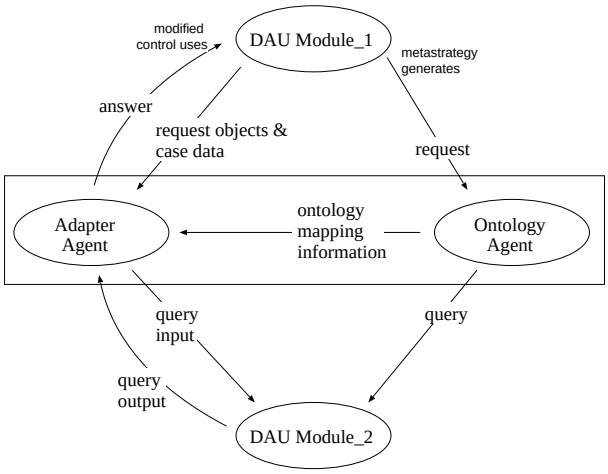

Figure 7: *Mediated Cooperation Scenario (slightly adapted from Fensel (1997))*

complex multi-stage analyses. One good example for such a problem is Natural Language Processing, another field the industrial relevance of which rapidly increases.

A main topic of this paper was the construction of a Document Analysis Core Ontology (DACO) as the fundament and the guideline for the collaboration of Document Analysis and Understanding modules. Issues derived from the DACO comprise inter-module aspects (communication between modules), as well as intra-module aspects (conditions on the reasoning within the modules) of collaboration. Without preparing modules along the ideas of the intra-module considerations, an ontology-supported cooperation scenario is useless.

Our previous C++ implementation [8] as well as Fresco and DSL (cf. Table 1) have already shown the benefits from hard-coded procedure calls during parsing. The use of strategic switching can be seen from Graphein; the use of declaratively encoded switching can be seen from Abney's "easy first strategy" [1]. Strategic switching was also simulated in our C++ implementation with coherently tuned grammars and multiple runs.

Our current implementation is in Perl, a programming language which eases the implementation of process intercommunication and graphical user interfaces. The first tests with the examples from [8] show that it is more efficient and convenient to work with this implementation. For example we used the possibility to interrupt the parser and counterchecked matches.

For setting up applications, a number of details need to be further elaborated. This paper delivered an ontological skeleton to start such an elaboration from. DACO, the Document Analysis Core Ontology, was provided. It serves as a common basis for the communication between research groups (e.g., to harmonize their systems). But moreover, DACO is the skeleton to be extended towards a DAU application ontology on the basis of which knowledge-based applications can communicate.

Our future work is devoted to three main issues.

For demonstrating the feasibility of our cooperation approach, we already prepared the DREAM system as suggested in this paper. The extension and implementation of more and more different functionalities will be a major task in the near future.

We collect agreed concepts in order to extend the core ontology skeleton and fill it towards a detailed, comprehensive, and reusable ontology of the DAU domain.

For the new concepts we will investigate the DAU methods and tasks which are applied to them; the methods to analyze instances of the concept, the methods to propagate information from analyzed instances to other concepts, and so on. When concepts and tasks are solicidated, a communication protocol for information exchange will be defined, and a meta-knowledge base describing the available modules will be built.

References

[1] S. Abney. Partial parsing via finite-state cascades. In *Proc. of the Robust Parsing Workshop, 8th European Summer School in Logic, Language, and Information, Prague*, 1996.

[2] St. Baumann, M. Malburg, H. Meyer auf'm Hofe, and C. Wenzel. From paper to a corporate memory — a first step. In *KI–97 Workshop on Knowledge-Based Systems for Knowledge Management in Enterprises*, Document D–97–03. Deutsches Forschungszentrum für Künstliche Intelligenz, 1997.

[3] A. Dengel and K. Hinkelmann. The Specialist Board - a technology workbench for document analysis and understanding. In M. M. Tanik, F. B. Bastani, D. Gibson, and P. J. Fielding, editors, *Integrated Design and Process Technology - IDPT96, Proc. of the 2nd World Conference*, Austin, TX, USA, 1996.

[4] D. Fensel. An ontology-based broker: Making problem-solving method reuse work. In *Proc. of the IJCAI-97 Workshop on Problem-Solving Methods for Knowledge-Based Systems*, Nagoya, Japan, 1997.

[5] Th. R. Gruber. A translation approach to portable ontology specifications. Technical Report KSL 92-71, Knowledge Systems Laboratory, Stanford University, April 1993.

[6] U. Jochum. Bibliothek, Buch und Information. *Bibliothek in Forschung und Praxis*, 15(3), 1991.

[7] B. Klein and A. Abecker. Ontological considerations for designing combinable modules in document analysis and understanding. 1998. Submitted for Publication.

[8] B. Klein and P. Fankhauser. Error tolerant document structure analysis. *International Journal on Digital Libraries*, 1(4), 1997.

[9] V. Lesser, B. Horling, F. Klassner, A. Raja, Th. Wagner, and S.XQ. Zhang. A next generation information gathering agent. In *Joint Proc. of the World Multiconference on Systemics, Cybernetics, and Informatics and the Int. Conf. on Information Systems Analysis and Synthesis*, 1998.

[10] A. Newell. The knowledge level. *Artificial Intelligence*, Vol 18., 1982.

[11] W. Umstätter. Schrift, Information, Interpretation und Wissen. *Bibliothek in Forschung und Praxis*, 16(2), 1992.

[12] M. Uschold and M. Gruninger. Ontologies: Principles, methods and applications. *Knowledge Engineering Review*, 11(2), June 1997.

[13] A. Valente and J. Breuker. Towards principled core ontologies. In *Knowledge Acquisition Workshop KAW'96, Banff, Canada*, 1996.

[14] G. van Heijst, A.Th. Schreiber, and B.J. Wielinga. Using explicit ontologies in KBS development. *International Journal of Human -Computer Studies/Knowledge Acquisition*, 1996. Special Issue on Using Explicit Ontologies.