An invoice — its semantics in the eyes of the beholder

Bertin Klein

DFKI GmbH, 67663 Kaiserslautern, Germany
bertin.klein@dfki.de

Abstract. A domain of semantic analysis is presented: invoices. This domain has been successfully tackled by us with a practical system. The analysis of scenario and system yield, first, ideas how to decompose a scenario or domain and, second, concepts resulting from the decomposition. It is then shown how to further develop the concepts found, and how to enter into a system design and implementation. This paper does not present implementation details nor screenshots (they have been presented elsewhere), and it does not discuss how our system is configured in detail for a given application. Instead, it shows a higher level, a "greatest common divisor" of semantic analysis projects. All results, ideas and concepts, are potentially useful also in other scenarios.

1 Introduction

The field of semantic annotation is wide, so we radically divided it first, and then conquered one (central and commercially interesting) division: the semantic annotation of invoices. A system successfully easing the job of people who process invoices shows, that this system successfully pinpoints and delivers the semantics of the invoices.[1] We have reviewed some systems for invoice reading [KD04b] and we have sold and installed a number of systems ourselves that process large volumes of incoming mail daily (i.e. invoices and everything else). [KD04a] Some let our system even completely autonomously account for money transfers. Systems, especially when they are large and commercially successful, seem to be fascinating. However, no system without an underlying model. "It requires a series of conceptual leaps to go from facts to a system design."[BH95] Here now, we want to share nothing but this underlying model. Note, that we usually do not distinguish between electronic invoices and scanned paper invoices. Second, the way we actually *annotate* the semantics depends on the application.

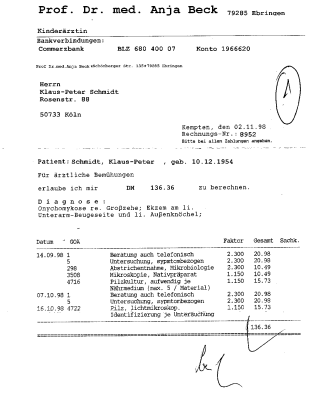
Semantic annotation requires an idea of the input —the spectrum of "things" to annotate that can occur— and of the output —the set of available annotations, i.e. meaningful concepts in a target domain. As announced, for the input, we restrict ourselves to invoices[2] (actually we use a single example medical invoice).

[1] To be precise: the semantics with respect to these people and their specific task.

[2] Our system disposes of a classification module, so that the invoice annotating module gets only invoices as input.

For the output, we commit to "the beholder", i.e. a typical receiver of an invoice, say in a company. The strategy is to look at a single instance of dealing with an invoice, prototypical for the scenario approached with our document analysis system. We analyze it and want to find out, what concepts a typical receiver has when dealing with the invoice. These concepts reveal what semantics this person seeks.

We have to admit, that our process of learning how (and once known, how easy it is) to filter out the important aspects took years: first years of scientific and prototypical work, and later years of projects in the field with (paying) customers. [KD04a]



Prof. Dr. med. Anja Beck 79285 Ebringen

Kinderärztin

Bankverbindungen:
Commerzbank BLZ 680 400 07 Konto 1966620

Herrn
Klaus-Peter Schmidt
Rosenstr. 88

50733 Köln

Kempten, den 02.11.98
Rechnungs-Nr.: 8952
Bitte bei allen Zahlungen angeben.

Patient: Schmidt, Klaus-Peter , geb. 10.12.1954

Für ärztliche Bemühungen

erlaube ich mir DM 136.36 zu berechnen.

Diagnose:
Conchomykose re. Großzehe; Ekzem am 11. Unterarm-Beugeseite und 11. Außenknöchel;

Datum	GOÄ		Faktor	Gesamt	Sachk.
14.09.98	1	Beratung auch telefonisch	2.300	20.98	
	5	Untersuchung, symptombezogen	2.300	20.98	
	298	Abstrichentnahme, Mikrobiologie	2.300	10.49	
	3508	Mikroskopie, Nativpräparat	1.150	10.49	
	4716	Pilzkultur, aufwendig je Nährmedium (max. 5 / Material)	1.150	15.73	
07.10.98	1	Beratung auch telefonisch	2.300	20.98	
	5	Untersuchung, symptombezogen	2.300	20.98	
16.10.98	4722	Pilz, lichtmikroskop. Identifizierung je Untersuchung	1.150	15.73	
				136.36	

Fig. 1. A typical invoice

2 Invoice processing step ontology

What somebody needs from a document, determines the semantics he needs. Let somebody get the invoice in Figure 1 delivered onto his desk. What is required to do? We elicited the following differentiation into eleven different steps (not all of them mandatory). This version elaborates on the version published in [DNW+03]:

1. *Classification.* Determining the class "invoice" triggers the processing steps decribed here. Further subclassing often makes sense.
2. *Retrieval of contract.* The patient needs to be spotted. Consider: addressee, firstname, lastname, birth date. A contract holder, or contract number could be specified, or a relation of patient to contract holder be mentioned ("son").
3. *Retrieval of operation.* Is there a folder with previous documents. Find the treatments and their dates in the invoice, the diagnosis. Conclude possible histories. Compare the (recent or timely related) activities which are logged with the contract (for the patient at hand).

4. *Justification of contract plausibility.* Do invoice address and contract address match? Are treatments admitted treatments, and took place while contract existed?
5. *Decide payment target.* Compensate the patient (if he already paid the doctor), the doctor or the doctors invoicing agency. What is: default, general rule in the company, usually better? Does the invoice have a due date, or other time constraints (so that the default rule is modified)?
6. *Collect payment data.* Spot account and bank (name and code of both), a reference number, a payment due date (permitting a delayed payment), comments (perhaps treatments or patient name or dates). Possibly extra comments are required, to explain reductions, etc. ...?
7. *Justification of payment data plausibility.* Is the location of the bank related to the recipient address? Do account name and code match with the data from the contract or possibly a company database of doctors. Does the reference number look genuine (i.e. "2313-AMB-12-1998" could raise doubts, because 1998 looks like a reference to the long ago year 1998)?
8. *Justification of invoice correctness.* Many invoices are not correct! Are all required information fields there, for invoices in general (like adressee, "trading good", tax number of invoicing party etc.), and for medical invoices (25% reduction for hospital treatments, ICD diagnosis code, ...). Are calculations correct? Are balances carried over correctly? Are treatment codes correct and combinations allowed?
9. *Justification of invoice plausibility.* Do treatments coincide and suit the diagnosis? Is the sequence of treatments plausible?
10. *Pay.* Retrieve the amount (claimed amount minus possibly reductions). Fill out bank name and code, account name and code. Provide the invoice-nr and other required reference in the comment field. Provide further comments. Set the due date. Commit.
11. *Archive and log.* Write a line in the contract log. Open or continue an/the operation. Store the invoice. Store the data read. Document the payment. Document and explain any non-standard actions.

Note, that real business operation on many invoices and large sums [KD04a] showed, that there were never problems with non-standard directives occuring in invoices. Some companies even let our system process many invoices without any individual human control.[3] At this point, it would be possible to construct a system (for paper invoices, only some available OCR tool needs to be added). However, we escort the reader one step further.

3 Building blocks of semantics of invoices

The software engineers who coded our system started their work with descriptions alike the invoice processing step ontology. On a closer look, all operations

[3] The system simply needs to commit less —less expensive— errors than human agents.

for the annotation of the semantics of an invoice sum up to not much more than a perhaps great but finite number of different comparisons. Only for additional (even minor or uncritical) convenience purposes a couple of scripts are required, to order dates, to calculate distances, and to check for "correct calculations" and correct "carry overs".

1. *Enumerable concepts:* A couple of concepts in invoices can be recognized with a database[4] with all their different possible values, e.g. "country := Ireland, Great Britain, Germany, ...". Examples: treatment, a patient (usually firstname, lastname, date-of-birth), a diagnosis, location of bank, a doctors address, treatment code. Actually we frequently use e.g. spotting of peoples names (including permutations, spelling errors, and possibly OCR errors), which is still very quick with databases of 1 Million names.
2. *Record concepts:* Another kind of database, often available in companies anyhow, is useful to conclude from facts to other facts, like the contracts DB. The annotation system can retrieve all "contracts" and get their "contract data", i.e. contract number, contract holder, operation, contract duration, contract address, contract coverage (which treatments). Other DBs could cover "treatment suitabilities for diagnoses", and "possible histories (from treatment to diagnosis to cause)."
3. *Visual databases:* Sometimes it is convenient to work with a (quick) database of 2 dimensional layout templates (abstracted layouts), to re-recognize known layouts, for which one can then rely on rather specific knowledge. This helps exploit that often invoicing parties use fixed layouts that are built-in in their computer systems.
4. *Labelled concepts:* A group of concepts are (most often) trivial to identify by a keyword indicating their meaning, like: "Diagnosis: flu". Their annotation requires a DB with the keywords (sets of synonyms). Examples: claimed amount, the patient, the diagnosis.
5. *Syntactic concepts:* Some constructs are conveniently described with regular expressions or comparable means of abbreviated, generative description. Typical examples are letter date, date of a treatment, payment due date. Also addresses are often described syntactically: address, the invoice address, addressee, doctors address. In early applications we described tables of treatments also syntactically. We subsume another group of concepts here, characterized by their layout, like subject and reference-field. However, the reference-field is determined also with a clear reference to its structure. Mostly, only the subject is spotted solely by position and bold font.
6. *Secondary concepts:* Secondary concepts [Sum98] are those, which can be identified only after other elements have been identified. Anyhow, the strategy is simple and good. After spotting the address, no other adress has to be searched, and the addres needs not be be considered in other searches.

Now, our software engineers started implementing, by filling up these building blocks with algorithms and glueing them together.

[4] Please read "ontology" instead of "database" from now on, if you know how to store knowledge in ontologies.

4 Conclusion

We have presented the building blocks to construct systems for the semantic annotation of invoices. The respective scenario was characterized with an informal ontology. We have reported about a commercialization (still nourishing 30+ people). Thus, first, the concepts presented in this paper, were already useful, are thus expedient. Thus, further, they carry some understanding of what the target user needs. This is valuable because it is hard to get. Users dont know what they do, not to mention, what they need. [BH95] (We fully agree. It took a long effort to get it.) Second, we carved out a division of semantic annotation, with two properties: commercial significance, plus, we were able to create a tool with reasonable annotation quality. Other fields meeting both these properties —enablers of achievement— can be carved out analogously. Third, we were able to use the presented building blocks to coordinate and align all the individual minds involved with the system (software engineers and all others).[5]

Already in [KA99] we had claimed, that with a then called "document analysis core ontology", furnished with some algorithms, document analysis systems can be constructed. Now our arguments are supported with evidence. We mapped all concepts on invoices to classes and gave an idea how to identify them, and how to bring them to life in an application. We think, and claim again, that the presented building blocks can be re-arranged and used to create on top of them systems to annotate the semantics of documents other than invoices.

References

[BH95] H. Beyer and K. Holtzblatt. Making customer-centered design work for teams. *Communications of the ACM Issue ( )*., May 1995.

[DNW+03] A. Dengel, P. Nowak, C. Wagner, K. Rehders, B. Klein, D. Schneider, M. Winkler, and Tebel R. Studie automatisierte rechnungseingangsbearbeitung marktpotential, marktübersicht und trends. commercial study, September 2003.

[KA99] B. Klein and A. Abecker. Distributed knowledge-based parsing for document analysis and understanding. In *IEEE International Conference about the Advances in Digital Libraries (ADL)*, May 1999.

[KD04a] Bertin Klein and Andreas Dengel. Problem-adaptable document analysis and understanding for high-volume applications. *International Journal on Document Analysis and Recognition*, 2004.

[KD04b] Bertin Klein and Andreas Dengel. Results of a study on invoice-reading systems in germany. In *IAPR International Workshop on Document Analysis Systems*, 2004.

[Sum98] K. Summers. *Automatic Discovery Of Logical Document Structure*. PhD thesis, Cornell University, 1998.

[5] "[..] the quality of the system is the result of the coordination of many individual actions. A whole team needs to develop the same vision for the system. The complex interplay between people needs to be managed." [BH95]