Towards multimodal public information systems

Magnus Merkel & Arne Jönsson
Department of Computer and Information Science
Linköping University

1. Introduction

In the future e-Home, information from various sources, located both globally and locally, are at hand for a wide range of tasks. Many of these tasks involve finding out about public authorities' rules and regulations. The Public Tax authorities, for instance, provide hundreds of documents on their web site (forms, FAQ's, tax rules, etc.). Currently, the user is restricted to navigating and searching these information sources by clicking hyperlinks or typing in keywords in a search box.

Suppose a citizen needs to know what the local tax in his area is. By providing the keywords "kommunalskatt" (local tax) and "Linköping" to the search engine five documents are retrieved and the user can continue clicking on the provided links to see if the answer is provided in the documents found. On the other hand, supposing that the user had the ability to state the information problem in natural language, and the background system was something more than just a document retrieval system, this interaction could look like the following:

Citizen:	Hi, how much do I pay in local taxes in Linköping?
System:	Are you a member of the Swedish Church?
Citizen:	Yes, I think so.
System:	What parish do you belong to?
Citizen:	Slaka.
System:	You pay 31,55 per cent in local taxes.

Figure 1. Sample dialogue

In order to allow for such interaction a number of research issues must be addressed. From a language technology perspective we foresee a fruitful, and necessary, co-operation between two main application areas:

- **Multimodal interaction**. This means that the user and system can utilise various types of modalities in order to present information, not only natural language (spoken or written) but also graphics, images, videos, and tables. In this paper we will, however, only consider natural language processing aspects, and especially dialogue and domain knowledge management. In the scenario in Figure 1, above, the use of dialogue allows formulation of the information needs in a fragmented fashion. The system had to collect further information before it could present something useful and this also often involves clarification sub-dialogues. When such a request has been completed, it can be used to retrieve the required information.
- **Information processing of documents.** In this paper we use *information processing* to mean interpretation and adaptation of information stored as natural language documents. Dialogue systems need structured information in order to support advanced information retrieval and problem solving. Such structured knowledge bases are often hand crafted. For the vast amount of public information available on the Internet it is not feasible to manually create structured knowledge sources. Instead, we must be able to find the relevant information stored in unstructured formats and in a systematic way convert it to a suitable form. The problem is not only to bring structure to the information, a prerequisite for doing that is also to locate the relevant information, which, as the information is unstructured is a complex task.

In a newly-started project at Linköping University we are investigating and developing multimodal interaction systems which utilise knowledge and methods from these two areas of language technology, see Figure 2. The long-term vision is to integrate such systems within a common e-home framework, but we believe that a fruitful research strategy will start from specific examples, and work towards a common framework, instead of trying to develop such a framework in a top-down fashion.

We will work iteratively based on a method that unifies issues of conceptual design with a clear correspondence to the components of the customisation of a generic framework (Degerstedt & Jönsson, 2001). The method advocates that coding and design go together and that a dialogue system is implemented iteratively with cumulatively added capabilities. Coding should be carried out as soon as possible, before all details of the system's design are ready. A prototype is developed from the start and is gradually refined based on evaluations of its behaviour. Furthermore, dialogue systems are knowledge intensive and much knowledge is acquired during the development of the system. The evolutionary development, thus, mainly involves refining the knowledge sources.

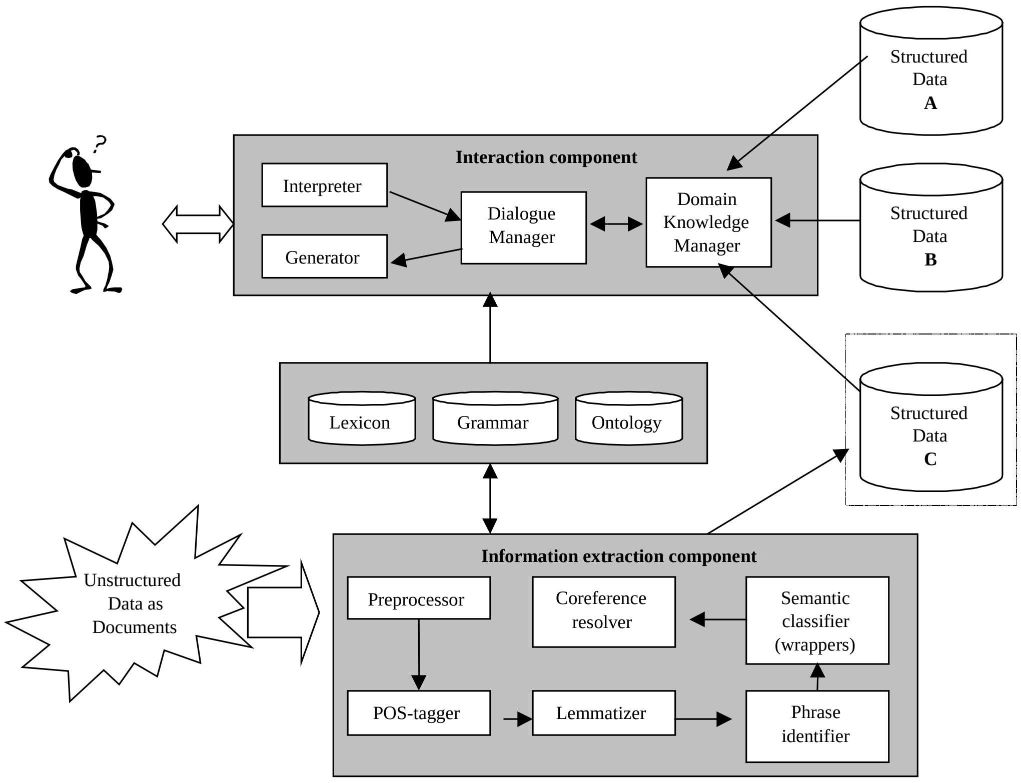

Figure 2. Interaction and information extraction combined

2. Interaction component

The interaction component interprets user utterances in context, access the background information system[1], integrate that with the interpreted utterance and generate a response. In this project the interaction will be handled by the MALIN dialogue system framework (Degerstedt & Jönsson, 2001). Dialogue systems often have a modular architecture with processing modules for interpretation, dialogue management, background system access, and generation, see Figure 2. The processing modules utilise a number of knowledge sources, such as, grammars, lexicons, a dialogue model, a domain model, and task models.

2.1 The Interpreter

The parser is an incremental chart parser that is modified to handle a grammar with rules that allow a partial and shallow parsing (Jönsson & Strömbäck, 1998) The interpretation is driven by the information needed by the background system and guided by expectations from the dialogue manager. The analysis is done by parsing as small parts of the utterance as possible.

Partial interpretation is particularly well-suited for dialogue systems, as we can utilise information from a dialogue manager on what is expected and use this to guide the analysis. Dialogue management also involves focus tracking as well as handling clarification subdialogues to further improve the interaction.

The lexicon is the main knowledge source for the parser. Fortunately, much information included in the lexicon can be acquired automatically from the information extraction component. Furthermore, as the grammar is based on partial information, many auxiliary words are not needed, which also makes automatic extraction and lexicon development easier.

2.2 The Dialogue Manager

The role of a dialogue manager differs slightly between different dialogue system architectures, but its primary responsibility is to control the flow of the dialogue by deciding how the system should respond to a user utterance. This is done by inspecting and contextually specifying the information structure produced by the interpretation module. If some information is missing or a request is ambiguous, clarification questions are specified by the Dialogue Manager and posed to the user. Should a request be fully

[1] In Figure 2 the background information system is depicted by the three knowledge sources to the right termed Structured Data A, B and C. This reflects the distributed nature of background information sources that is typical for many of the information systems addressed in this work.

specified and unambiguous the background system can be accessed and an answer is produced. As a basis for these tasks the Dialogue Manager can utilise a dialogue model, a system task model, and a dialogue history.

The Dialogue model holds a generic description of how the dialogue is to be constructed, i.e. to decide what action to take in a certain situation. It is used to control the interaction, which involves determining: 1) what the system should do next (and what module is responsible for carrying out the task) and 2) deciding what communicative action is appropriate at a given dialogue state.

The Dialogue model utilised in MALIN is the LINLIN model (Jönsson, 1997), which is a structurally based model that uses a dialogue grammar for dialogue control. The dialogue is structured in terms of discourse segments, and a discourse segment in terms of moves and embedded segments. Utterances are analysed as linguistic objects, which function as vehicles for atomic move segments. An initiative-response IR structure determines the compound discourse segments, where an initiative opens the IR-segment by introducing a new goal and the response closes the IR-segment. The LINLIN dialogue model classifies the discourse segments by general speech act categories, such as "question" (Q) and "answer" (A), rather than specialised (cf. Hagen 1999), or domain related (Alexandersson and Reithinger, 1995).

The dialogue segments form a dialogue tree. The nodes in the tree, termed dialogue objects, hold information such as the current objects and properties, the user request in focus, information on speaker, hearer, type of general speech act, etc. The dialogue tree is naturally specified in terms of a grammar.

The model assumes that decisions as to what to do next are made on the basis of focus information (cf. Leceuche et al. 2000), i.e. depending on how the focal parameters have been specified. Focus information can be copied between nodes in the dialogue tree, either horizontally by "focus inheritance", from one IR-segment to the next, or vertically by "answer integration" to handle sub-dialogues.

To develop a Dialogue Manager that easily can be customised to new domains and in which different dialogue strategies can be explored, the Dialogue Manager should only be concerned with phenomena related to the dialogue with the user. It should not be involved in the process of accessing the background system or performing domain reasoning. These tasks should instead be carried out by a separate module, a Domain Knowledge Manager.

2.3 The Domain Knowledge Manager

One novel feature of our architecture is the Doamin Knowledge Manager (Flycht-Eriksson, 2000). This module makes the MALIN framework especially suitable for unstructured domains.

The primary responsibility of the Domain Knowledge Manager is to provide domain- and application-specific information when the Dialogue Manager has produced a fully specified request. The Dialogue Manager can deliver a request to the Domain Knowledge Manager and in return expect to get the requested information or a motivation of why the information could not be retieved. The Domain Knowledge Manager retrieves and integrates information from application information sources utilising the domain ontology. If the Domain Knowledge Manager encounters a problem it cannot solve by using its knowledge about the domain, a specification of the problem and the needed clarifying information is returned to the Dialogue Manager.

Access of application information sources can be problematic in two ways: no answer to the request can be produced, or too many answers are found. The first situation can occur if a request is inconsistent or if no object meets all the restrictions of the request. Two different approaches to dealing with this are for the system to try and fix it itself, or for the system to help the user to handle the situation. The first approach includes relaxing some of the constraints or resolving the inconsistency, both of which require reasoning about the domain. If the system fails or does not try to solve the problem itself, it can give the user as much help as possible when dealing with the problem, for example by stating the cause of the problem and suggesting how the request should be modified. The MALIN framework supports both approaches.

The second problematic situation, in which a request has resulted in too many answers from the background system, can arise from requests that are not specific enough. The solution chosen in the MALIN framework is to use the domain knowledge, manifested in the domain ontology, and decide which constraints should be asked for in order to specify the request, thus helping the user to formulate a more specific request.

3. Information Extraction component

The purpose of the information extraction component is to transform information from unstructured or semistructured document collections into structured information, in the form of a *document warehouse* where information from multiple document types and from multiple sources are stored (Sullivan 2001). The document collection for specific domains from a public agency, for instance, are most often found in various formats (html, pdf, xml and different office program formats) and they come from various sources such as multiple network servers, intranets and the Internet. The solution to the conversion from unstructured to structured information is to be found among techniques from the field of information extraction (Cowie & Lehnert 1996), basically a mix of string-based information retrieval and shallow linguistic approaches. The shallow linguistic analysis, such as generating wrappers (or agents) are used to identify entities such as names of people, prices, countries, etc, which provides a way of extracting certain sets of simple facts from the documents. Mattox et al. (1999) argue that many IE approaches based on shallow parsing will yield insufficient structure if the extracted information is to serve as a database to answer more elaborate and "important" queries. By important queries, they mean, for instance, questions about something that is not explicitly expressed in the document. If a document contains information about the national debts of all countries in absolute figures, then a more elaborate system should be able to answer queries about which countries that have a debt that *exceeds* X billion dollars.

There are several ways of identifying concepts and relations between objects in unstructured documents in a step-by-step manner:

1. Pre-processing (unveiling document formats such as headings, table structure, lists, etc).
2. Parts of speech tagging
3. Lemmatization
4. Phrase clustering
5. Semantic classification
6. Discourse reference identification
7. Identification of relationships between concepts
8. Output generation

Step 1 involves non-linguistic information and is done to normalise the documents into a common format that the rest of the machinery can build on. Step 2 through 4 requires syntactic and morphological taggers that can provide syntactic information for the domain(s) at hand. Step 4 can also be enhanced by statistically-based phrase extractors such Frasse-II (Merkel & Andersson 2000). Step 5 requires some knowledge about the domain and application at hand, usually expressed as a domain specific ontology. In the Message Understanding Conferences (MUC, e.g. described in Grishman & Sundheim, 1996), there has been strong encouragement to measure aspects of the internal processing of extraction systems and how well they solved problems of coreference, word sense disambiguation and providing predicate-argument structure for a particular segment of text and sentence. The latter aspects all have bearing on steps 5 to 7 in that they pinpoint the need for a deeper understanding of content and relationships.

4. Pilot study

In this project we intend to develop wrappers that identify concepts from the domain documents in the tax domain. Initial investigations on how such wrappers can be developed and used were carried out during the spring of 2001.

Four student groups in Linköping were assigned the task of building a questioning-answering system in the domain of Swedish birds. One of the subtasks was to analyse a set of documents that contained information about roughly 100 Swedish birds and to use various approaches to identify entities and relationships in these unstructured documents. The structured bird document base was then to be connected to a natural language interface which the students had to write from scratch together with a module that should provide an answer to the user's question; thus enabling the user to search for birds using natural language. Here are some examples of the questions that the Bird system were planned to handle:

1. What is the latin name for magpie?
   Answer: Pica Pica
2. A bird that is greyish brown, yellow and has yellow and black streaks on the head and seen in pine forests?
   Answer: Goldcrest
3. Is a raven larger than a crow?
   Answer: Yes.

Figure 3. Sample dialogue from the Bird system

Semantic information was extracted from the enriched documents based on the XML-tags and some of this information was then added to the lexicon which was used by both the question interpreter and the search module. Some semantic knowledge was definitely hand-coded and some of the wrappers were not very generic, but given the time and situation, the solutions provided were nevertheless very encouraging.

The experiences from the Bird task gave many insightful experiences to information extraction. First, it was shown that five-six people could actually build a system from scratch with a small set of resources in two weeks time. Secondly, given a pre-made package for handling the interaction part, a great deal of more work could have been put into the semantic classification (the wrapping module), which would have meant that a more generic solution could have been adopted from the start. None of the groups did for instance use any explicit domain ontologies.

In the future system, depicted in Figure 2, the Extraction component is sketched. Some of the linguistic resources are shared with the Interaction component, for example, the lexicon and the ontology. When new concepts are being identified during the extraction process, the lexicon will (automatically) be updated and thereby increase the interaction component's ability to correctly interpret the user's questions.

5. Summary and future work

In this paper we have briefly presented current work on the development of a system that utilise information extraction techniques in a dialogue system. A number of issues has not been elaborated upon in this short paper, such as how to utilise various modalities, how to generate usable responses and how to integrate and interpret multi-modal interaction, instead focus has been on issues related to how and where dialogue systems meet information processing.

One important feature of our dialogue system's architecture is the use of a domain knowledge manager. The domain knowledge manager acts as the intermediator between the two sub-areas: information processing and multi-modal interaction. We have previously discussed the advantages of a separate module for domain knowledge management (Flycht-Eriksson and Jönsson, 2000). One such advantage is that the dialogue manager need not be aware of how the domain is structured and need not consider such issues as part of the contextual interpretation of a user request. A separate domain knowledge manager also means that the information processing components need not consider aspects of dialogue.

Common knowledge sources need to be developed, the properties of which have to be established. This also includes issues on the parts that are shared among the dialogue system and the information processing modules. Presumably, the ontology is a common resource, and so is the lexicon, whereas the grammars probably differ.

The development of a multi-modal information system that integrates a dialogue system with techniques from information extraction will, as stated above, be carried out iteratively from simple prototypes towards more complex dialogue systems. Thus, we will always have a working system whose capabilities can be incrementally augmented as the system evolves, e.g. while the ontology for the whole domain is being expanded.

References

Jan Alexandersson and Norbert Reithinger, "Designing the Dialogue Component in a Speech Translation System", Proceedings of the Ninth Twente Workshop on Language Technology (TWLT-9), pp 35-43, 1995.

Jim Cowie & Wendy G. Lehnert. Information extraction. *Communications of the ACM,* 39(1):80-91.

Lars Degerstedt and Arne Jönsson, "A Method for Iterative Implementation of Dialogue Management", 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems, August 5, 2001, Seattle

Annika Flycht-Eriksson, "A Domain Knowledge Manager for Dialogue Systems", Proceedings of the 14th European Conference on Artificial Intelligence (ECAI), IOS Press, Amsterdam, 2000.

Ralph Grishman & Beth Sundheim. Message Understanding Conference - 6: A Brief History. In *Proc. of the 16th Int'l Conf. on Computational Linguistics, Copenhagen*, 1996.

Eli Hagen, "An approach to Mixed Initiative Spoken Information Retrieval Dialogue", User modeling and User-Adapted Interaction, Vol. 9, No. 1-2, pp 167-213, 1999

Arne Jönsson, A model for habitable and efficient dialogue management for natural language interaction, Natural Language Engineering 3(2/3), pp 103-122, Cambridge University Press, 1997.

Arne Jönsson & Lena Strömbäck Robust Interaction through Partial Interpretation and Dialogue Management, Proceedings of Coling-ACL'98, Montrèal, Canada, 1998.

Renaud Leceuche, Dave Robertson, Catherine Barry and Chris Mellish, "Evaluating focus theories for dialogue management", International Journal on Human-Computer Studies, Vol. 52, pp 23-76, 2000.

David Mattox, L. Seligman & K. Smith. Rapper: A Wrapper Generator with Linguistic Knowledge. In *2nd Proceedings from the Workshop on Web Information and Data Management,* Kansas City, Missouri, pp. 6-11, 1999.

Magnus Merkel & Mikael Andersson. Knowledge-lite extraction of multi-word units with language filters and entropy thresholds. In *Proceedings of RIAO'2000,* Collége de France, Paris, France, April 12-14, 2000, Volume1, pp. 737-746, 2000.

Dan Sullivan. *Document Warehousing and Text Mining,* John Wiley & Sons, 2001.