

A Multimodal Conversational Companion for Reminiscing about Images

Roberta Catizone, Simon Worgan, Yorick Wilks , Alexiei Dingli and Weiwei Cheng

{r.catizone; s.worgan; y.wilks; w.cheng}@dcs.shef.ac.uk, alexiei.dingli@um.edu.mt

Introduction

COMPANIONS is a project that aims to change the way we think about the relationships of people to computers and the Internet by developing a virtual conversational 'Companion'. This will be an agent or 'presence' that stays with the user for long periods of time, developing a relationship and 'knowing' its owners preferences and wishes. It will communicate with the user primarily by using and understanding speech, but also using other technologies such as touch screens and sensors.

This paper describes the functionality of, and immediate plans for, the Senior Companion (SC), one of two initial prototypes built in the first eighteen months of the project, and on whose evaluation the second half of the project (2008-2010) will be based. The Senior Companion focuses on communicating multimodality with the old, and linking them to the wealth of stored digital life information they have but may find difficult to organize. The Companion will, through conversation, elicit their life memories, often prompted by discussion of their photographs; the aim is that the Companion should come to know a great deal about its user, their tastes, likes, dislikes, emotional reactions etc, through long periods of conversation. It is a further assumption that most life information will be stored on the internet (as in the Memories for Life project, REF) and we are linking the SC directly to photo inventories in Facebook (see below). One could see the overall aim of the SC as producing a coherent life narrative for its user from these materials, although its short term goals are to assist, amuse, divert and gain the trust of the user.

The technical content of the project is use a number of types of machine learning (ML) to achieve these ends in original ways, initially using a methodology used by the Sheffield University Companions group in earlier research: first, by means of an Information Extraction (IE) approach to deriving content from user input utterances; secondly, using a training method for attaching Dialogue Acts to these utterances (Webb, et al.) and lastly, using a specific type of dialogue manager (DM) that uses Dialogue Action Forms (DAF) to determine the context of any utterance, and a stack of these DAFs as the virtual machine that runs the dialogue and models by means of shared user and Companion initiative (REF). The SC is not a robot and could be embodied in a screen, a handbag or a mobile phone %% while retaining the same "personality": it is more a very high level internet agent, dedicated to a single user over a long term. In what follows we shall:

1. describe the current SC prototype's functionality;

2. sketch its architecture and modules, focusing on the Natural Language Understanding module and the Dialogue Manager.
3. set out our short term plans to enhance Dialogue Management performance with direct Internet access and initial ML experiments.

The Senior Companion System

The Senior Companion (SC) prototype (Wilks 2007, 2008; Wilks et al., 2008) is designed for quick advance over the first project year so as to be basis for a second round of prototypes embodying more advanced ML; it contains the functionality described in what follows, which includes news and photographs.

The central function of the SC is engaging the user in discussion about their photographs : where and when they were they were taken, details about the people in them and their relationship to the user and each other. The SC keeps a record of the user's input and is able to pick up discussion with the user where the system left off in later user sessions. In addition to reminiscing, the SC also allows the user to do basic photo management including photo selection by pointing and grouping of photos through dialogue.

Once a photo is loaded, it is processed with face recognition software to identify any faces in it. This recognition software, OpenCV, provides positional information by identifying the face coordinates and this information is exploited in the Dialogue Manager by making reference to the position of those in the photograph (person on the left, right, center, etc.) as well as recognising when there are groups of people. The system discusses properties of the photo as well as properties and relationships of the people in the photos.

The news reading feature adds an interesting accompaniment to the photo domain and demonstrates the ability of the system to handle more than one kind of application at a time, and has, of course, an unconstrained vocabulary . The news, taken via RSS feeds from the BBC news website, includes news from three popular categories : politics, sports and business, and the user can choose between them, stop and start the feed by speaking etc.

This basic system provides the components for rapid future development of the SC, as well as its main use as a device to generate more conversation data for machine learning research

in the next project phase. The system architecture and modules are described briefly below.

The Senior Companion:

- Includes a visually appealing multi-modal interface with a character avatar to mediate the system’s functionality to the user.
- Interacts with the user using multiple modalities – speech and touch.
- Includes face detection software for identifying the position of faces in the photos.
- Accepts pre-annotated (XML) photo inventories as a means for creating richer dialogues more quickly.
- Has the ability to engage in conversation with the user about topics within the photo domain: when and where the photo was taken, discussion of the people in the photo including their relationships to the user.
- Contains a feature for reading news from three categories: politics, business and sports.
- Contains a feature for telling jokes taken from an internet-based joke website.
- The system retains all user input for reference in repeat user sessions, in addition to the knowledge base that has been updated by the Dialogue Manager on the basis of what was said.
- Contains a fully integrated Knowledge base for maintaining user information which contains:
 - Ontological information which is exploited by the Dialogue Manager and provides domain-specific relations between fundamental concepts.
 - A mechanism for storing information in a triple store (Subject-Predicate-Object)-the RDF Semantic Web format---- for handling unexpected user input that falls outside of the photo domain, e.g. arbitrary locations in which photos might have been taken.
- Contains a reasoning module for reasoning over the Knowledge Base and world knowledge obtained in RDF from the Internet.
- Contains basic photo management capability allowing the user in conversation to select photos as well as display a set of photos with a particular feature.

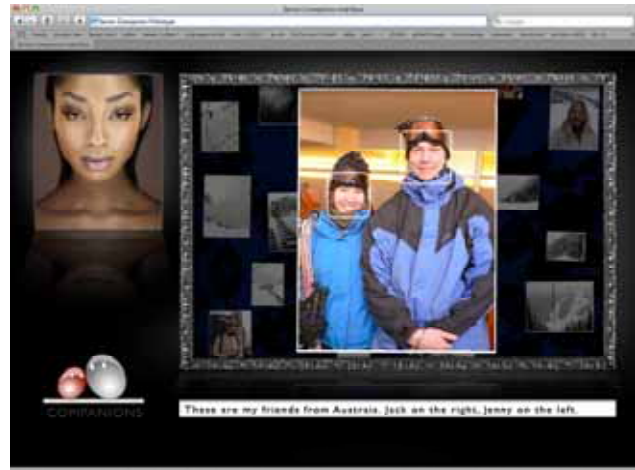


Figure 1 : Senior Companion Interface

System Architecture

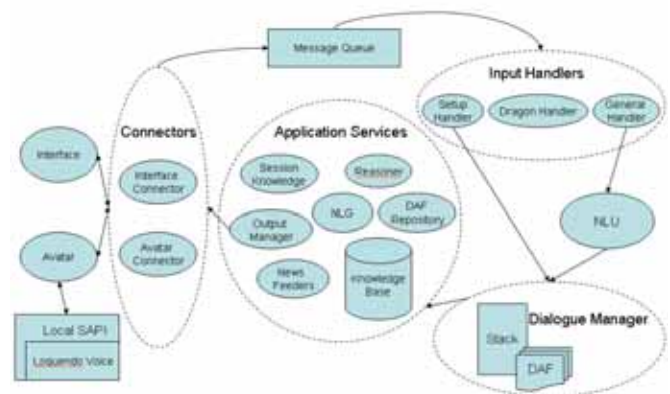


Figure 2 : Senior Companion system architecture

The above figure is the Senior Companions architecture. It contains three abstract level components – Connectors, Input Handlers and Application Services - plus the Dialogue Manager.

Connectors form a communication bridge between the core system and external applications. The external application refers to any modules or systems which provide a specific set of functionalities that might be changed in the future. There is one connector for each external application. It hides the underlying complex communication protocol details and provides a general interface for the main system to use. This abstraction decouples the connection of external and internal modules, makes changing and adding new external modules easier. At this moment, there are two connectors in the system – Napier Interface Connector and CrazyTalk Avatar Connector. Both of them are using network sockets to send/receive messages.

Input Handlers are a set of modules for processing messages according to message types. Each handler deals with a category of messages where categories are coarse-grained and could

include one or more message types. The handlers separate the code handling inputs into different places and make the code easier to locate and change. Three handlers have been implemented in Senior Companions system – Setup Handler, Dragon Events Handler and General Handler. Setup Handler is responsible for loading the photo annotations if any, performing face detection if no annotation file is associated and checking the Knowledge Base if the photo being processed has been discussed in earlier sessions. Dragon Event Handler deals with dragon speech recognition commands sent from interface while the General Handler processes user utterances and photo change events of the interface.

Application Services are a group of internal modules which provide interfaces for the Dialogue Action Forms (DAF) to use. It has an easy-to-use high level interface for general DAF designers to code associated tests and actions as well as a low level interface for advanced DAF. It provides the communication link between DAFs and the internal system and enables DAFs to access system functionalities. Following is a brief summary of modules grouped into Application Services.

News Feeders are a set of RSS Feeders for fetching news from the internet. Three different news feeders have been implemented for fetching news from BBC website Sports, Politics and Business channels. There is also a Jokes Feeder to fetch Jokes from internet in a similar way.

DAF Repository is a list of DAFs loaded from files generated by the DAF Editor. A fresh copy of a DAF can be obtained by passing the DAF name to this module.

NLG is responsible for randomly selecting a system utterance from a template. An optional variable can be passed when calling methods on this module. The variable will be used to replace special symbols in the text template if applicable. For example, a template utterance “How do you know \$?” will be returned as “How do you know John?” if passing variable “John” when calling generation method of this module.

Session Knowledge is the place where global information for a particular running session is stored. For example, the name of the user who is running the session, the list of photos being discussed in this session, the list of user utterance and etc.

Knowledge Base is the data store of persistent knowledge. It is implemented as an RDF triplestore using a Jena implementation. The triplestore API is a layer built upon a traditional relational database. The application can save/retrieve information as RDF triples rather than table records. The structure of knowledge represented in RDF triples is discussed later.

Reasoner is used to perform inference on existing knowledge in the Knowledge Base.

Output Manager deals with sending message to external applications. It has been implemented in a publisher/subscriber fashion. There are three different channels in the system – the text channel, the interface command channel and the avatar command channel. Those channels could be subscribed by any connectors and handled respectively.

Natural Language Understanding

The purpose of the Natural Language Understanding module is to extract meaning from user utterances using a set of well-established Natural Language processing tools such as the GATE (Cunningham, et al., 1997) system. The basic processes carried out by GATE are: tokenizing, sentence splitting, POS tagging, parsing and Named Entity Recognition. These components have been further enhanced for the SC system by adding new and improved gazetteers. These include amongst others new locations and family relationships. The NE recognizer is a key part of the NLU module and recognizes the significant entities required to process dialogue in the photo domain: PERSON NAMES, LOCATION NAMES, FAMILY_RELATIONS and DATES. Although GATE manages to recognise basic entities, more complex entities are not handled. Because of this, apart from the gazetteers mentioned earlier, new extraction rules using the JAPE rule language found in GATE were also developed for the SC module. These included rules which identify complex dates, family relationships, negations and other information related to the SC domain. With this information, the NLU module identifies Information Extraction patterns in the dialogue that represent significant content with respect to a user's life and photos. The NLU module also identifies a Dialogue Act Tag for each user utterance based on the DAMSL set of DA tags and prior work done jointly with the University of Albany (Webb et al., 2008).

The SC Knowledgebase

The NLU module provides the DM with a myriad of information. This information, even though relevant, might not be useful for the domain. If we assume that the NLU module identified Tom who is the brother of Jack, this information is extremely important for the domain since we're dealing with family relations yet, the fact that Tom and Jack are both nouns (from a linguistic point of view) is still relevant, but not useful for the domain. Thus this information is discarded by the DM. When the DM filters this information, the SC knowledgebase comes into play since this information is stored inside it. The SC knowledgebase can be considered as being the long term storage of the system. Any information which might be useful by present or future interactions with the SC is stored inside the knowledgebase. This information can be retrieved anytime when needed. The structure of the KB is an RDF triple store. The triple store is a kind of database which allows for fast storing or retrieval of RDF triples where each triple is made up of a subject, a predicate and an object. So back to the example mentioned earlier, Tom would be the subject, brother would be the predicate and Jack would be the object as follows (Tom brotherOf Jack) which simply means that Tom is the brother of Jack.

In mathematical terms, the triple store is nothing more than a large database of interconnected graphs. So in the triple above, if we imagine it in a two dimensional space, Tom and Jack would be two distinct points which mark the beginning and end respectively of a line on the 2D space and the brother relationship would be the relationship (or line) that joins them in

that space. The first advantage of using this structure is that the relationships between different entities are explicitly defined via the predicates in the triples. The second advantage of this is that it is quite easy to perform inferences on the data. In fact, as part of the knowledgebase, we have a large set of inference rules which we use to infer new data. If we stick to the previous examples and add a new triple which states that Mary is the daughter of Tom (Mary daughterOf Tom), we can infer new data using the inference engine and the following inference rule:

```
[Niece1: (?a brotherOf ?b),
      (?x daughterOf ?a),
      (?x gender female) -> (?x nieceOf ?b)]
```

So in the previous rule, (Tom brotherOf Jack) matched with the first part of the rule, (Mary daughterOf Tom) matches with the second part of the rule and the gender part is a piece of information which is supplied to us by the NLU. Thus we get ...

```
[Niece1: (Tom brotherOf Jack),
      (Mary daughterOf Tom),
      (Mary gender female) -> (?x nieceOf ?b)]
```

After we run the inference engine, we get (Mary nieceOf Jack) thus adding new and previously unknown knowledge to our knowledgebase. This combination of triple store, inference engine and inference rules makes a powerful system which mimics human reasoning and thus gives the SC a sense of intelligence. For our prototype we are using the JENA Semantic Web Framework for the inference engine together with a MySQL database as the knowledgebase.

Dialogue Management

The Companions DM can be thought of as a simple virtual machine intended to capture a range of dialogue phenomena: topic exhaustion, topic reentry, the coherence of individual topics, the recursive hierarchical structure of subtopics and plans, separate initiatives on the part of user and Companion, as well as a procedural embodiment of the central notion of the focus-of-attention topic of a dialogue. This is a notion that information-state approaches express as QUD, the question under discussion, a difference that precisely captures the opposition we intend: we intend that such notions have direct procedural equivalents, easily demonstrated, as opposed to what one could call folk-reification, making a module or entity correspond to a theoretical object (a position immortally captured by McDermott in his (McDermott, 1976).

In the SC DM, we have a stack and the structure on top of the stack is the focus of attention or topic: such structural objects are called DAFs (Dialogue Action Forms) and are networks, intended to express the dialogue activity associated with a topic. A topic can be of any grain size, specialized to a particular person, e.g. Fred Jones, a place, Pisa, or the topic of asking about a specific photo, a DAF object that will be an instantiation of a general DAF for conversation about an image on the screen. A DAF can also capture a meta-topic that serves to regulate the dialogue, such as one covering stereotypical dialogue behaviour in capturing, and escaping from, misunderstandings.

The DAFs are in fact ATNs (Augmented Transition Networks) originally designed by Woods [Woods 1980] for syntactic analysis, structures that embody both hierarchical sub-networks (i.e. RTNs in automata theory) as well as augmentations on the arcs between nodes which enable any operations to be performed, within the dialogue process or offline in the knowledge base or multimodal world. Thus, the networks as dialogue controllers have effectively Turing Machine power although that fact has no practical meaning, and the proven advantages of ATNs are the global structures they display (in syntax or dialogue), as opposed to collections of unordered transition rules (cf. PSG parsers or, in dialogue, the Information State approach). Their use thus expresses the belief that, in dialogue, there are global topic structures, or mega- or meta-structures to be designed or learned, as opposed to firing matching rules from an inventory. There is of course an assumption behind this use of larger-scale structures that (a) here exists stereotypy in dialogue at this level, that we can capture, by hand or with some form of ML, and (b) we have a method for switching between DAFs when needed that overcomes the obvious drawback of being “stuck” in a DAF at the top of the stack when the user has in fact change the topic. The test of these hypotheses will be the performance of the system as a whole and the learnability of DAFs and heuristics for moving between them on the stack.

Dialogue Action Forms (DAFs) closeup

Communication between DAFs and the Core System

We have developed a GUI interface for designing DAFs which simplifies the task of writing DAFs (see example Figure 4). The DAFs contain nodes and arcs between nodes where the nodes represent system states and the arcs represent moving between states through tests and actions. The communication between DAFs and the core system happens in the actions and tests code of the DAFs. Figure 3 shows this link in the new architecture:

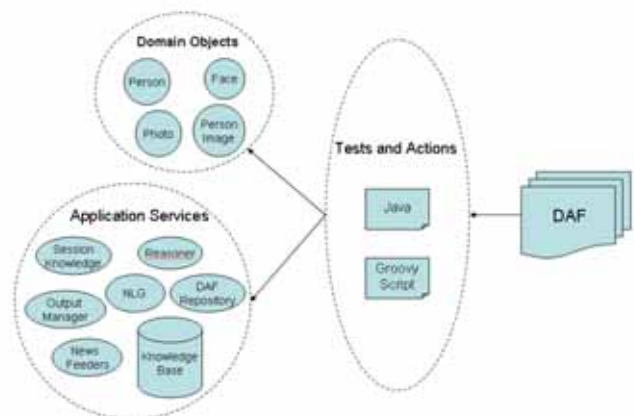


Figure 3: DAFs system communication diagram

In our system, there are two ways to implement a test and action – either using Java or using Groovy Script. The main difference

between these two approaches is the way they talk to the rest of the system. In Java, the code can see every public interface of each module since the core system is implemented in Java, while in Groovy Script, the code talks with the system through variable bindings. In either case, there are handy high level interfaces to use. It is also possible to use low level interfaces provided by individual modules to implement new functionality from scratch.

A variable binding is nothing more than a list of variables. It manages the variable type conversion between Java and Groovy.

The initial reason of bringing a scripting language in DAFs is to pass variables in actions and tests in order to reuse DAF structures. This goal has been achieved by setting local variables in the DAF. However, there are other benefits of using a scripting language in DAFs:

1. The script doesn't need to be compiled into binary code but is interpreted dynamically by the system. The other approach of attaching java code takes more effort as it needs to be compiled into binary code to be executed. There is also no overhead of setting classpath. The main issue with dynamically interpreting Java code is performance as Java wasn't designed as a dynamic language.
2. Scripting languages don't have strong syntax requirements and are type-free thus easier to understand and write. With a brief introduction and example, people who are not familiar with programming can start writing script. While writing java code is not trivial and needs more study.
3. As scripting language is easy to read and write, both DAF designers and system developers can write them. It helps the designers to gain more insight into the behavior and logic of the DAF and think holistically while designing the DAF. It also serves as a common communication language between DAF designers.
4. By grouping small pieces of script into larger chunks, new functionality can be added easily without changing or recompiling the core system. It also provides a future extension of the main system making it possible to extend the system functionality by developing more scripts. For example, reading in a file as part of a DAF action would be better implemented in a script rather than in the core system which would require redeployment of the system.

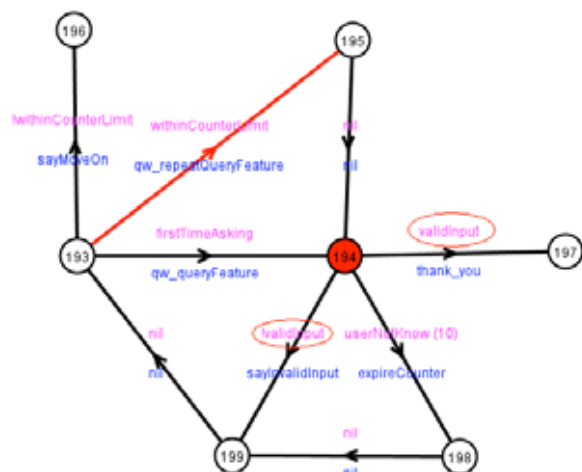


Figure 4: General DAF for validity checking

Tests and Actions

For each arc in a DAF, there is one test and action associated with it. The test is placed above the arc while the action is below the arc.

Nil Tests and Actions

A test or action with name "nil" (case-sensitive) means the test always returns true or action does nothing.

Arc priority

Each arc has a section of setting a priority value. If no prior value is set, the default value will be set as 0 by the system at run time. The current maximum value is 10. When examining the tests, if more than one arc meets the pass condition, the arc with higher priority will be selected.

Ensuring Persistence of Personality through Intelligent Adaptation

Traditionally, spoken dialog systems have been able to learn and reason over uncertainty through a partially observable Markov decision process (POMDP) that uses reinforcement learning (RL) to modify its strategy according to some reward function (Young, 2002). In restricted domains with a clearly defined goal (e.g. placing a pizza order) RL proceeds by offering a large reward when the task is successfully completed. In the open-ended scenarios proposed by the COMPANIONS project how do we capture this reward function? We propose that the Companion should be tasked with maintaining the users positive emotional state, ensuring continued user satisfaction. Given the goal of maintaining this state emotion becomes a central motivating factor requiring an internal representation of both itself and its user.

We considered two proposed representations of emotion, the discrete theory of basic emotion (Ekman, 1999) and the continuous theory, which maps a range of emotions onto a two dimensional space (Wundt, 1913, Cowie, Douglas-Cowie, Savvidou, & McMahon, 2000). We have settled upon the continuous theory, as instead of arbitrarily defining a number of discrete emotional categories we can attempt a data driven approach. By creating a self organising map (SOM) (Kohonen, 1982), capturing the variability of emotional speech, we can train the system on a wide range of emotional utterances forming distinct attractors within the defined emotional space. Having trained the system these attractors will then be tied to a subset of dialogue acts, reflecting the agents current emotional state. RL can then proceed by allowing the Companion to learn how its own movement in this space affects the user. However, this is only possible if the Companion can place a representation of the state of the user into this space, accordingly we need to ascertain the users current emotional disposition.

We propose to achieve this by extracting emotional information from a number of multimodal inputs. By combining the semantic content and acoustic information of the current utterance the uncertainty of the current emotional state can be reduced (Lee, Narayanan, & Pieraccini, 2002, Lauria, 2007). Having established an understanding of the users emotional state the Companion will then have a number of 'tools' with which to manipulate this scenario.

The Companion will converse with the user through a number of conversational levels, the tone and paralinguistic nature of the utterance can be adjusted to convey emotional content, the nature of the dialogue act can be similarly adjusted and the overall conversational strategy can be optimised. When exploiting these features it is important to maintain a consistent personality that accurately conveys the capabilities and limitations of the Companion; any gap between user expectation and reality would create an obvious source of user dissatisfaction. As shown by previous WOZ studies (Moore & Morris, 1992) simple differences to tone of voice or appearance can cause dramatic differences in user behaviour. Accordingly, the Companion will present a unified personality (encompassing appearance, voice, personality and behaviour) which accurately manages the user's expectation.

References

Cowie, R., Douglas-Cowie, E., Savvidou, S., McMahon, E., Sawey, M. and Schroder, S. (2000). 'FEELTRACE': An instrument for recording perceived emotion in real time. ISCA Tutorial and Research Workshop (ITRW) on Speech and Emotion, Newcastle, UK .

Cunningham et al. (1997). Cunningham, H., Humphreys, K., Gaizauskas, R., Wilks, Y., GATE -- a TIPSTER based General Architecture for Text Engineering. In Proceedings of the TIPSTER Text Program (Phase III) 6 Month Workshop. DARPA, Morgan Kaufmann, California

Ekman, P. (1999). Basic Emotions. Handbook of Cognition and Emotion, Dalglish, T. and Power, M. (Eds.) John Wiley, New York.

Lauria, S. (2007). Talking to Machines: Introducing Robot Perception to Resolve Speech Recognition Uncertainties. Circuits Systems Signal Processing vol 26(4) pp. 513-526.

Lee, Chul Min, Narayanan, Shrikanth S. / Pieraccini, Roberto (2002): "Combining acoustic and language information for emotion recognition", In *ICSLP-2002*, 873-876.

Moore, R., & Morris, A. (1992). Experiences collecting genuine spoken enquiries using WOZ techniques. Fifth DARPA Workshop on Speech & Natural Language Harriman, NY.

Song, M., You, M., Li, N., & Chen, C. (2008). A robust multimodal approach for emotion recognition. *Neurocomputing* .vol. 71 pp. 1913-1920

Webb, N., Liu, T., Hepple, M., and Wilks, Y. (2008). Cross Domain Dialogue Act Tagging. In Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC-08). Marrakech, Morocco, 2008

Wilks, Y. (2007) Has there been progress on talking sensibly to computers ? *Science*, Volume: 318

Wilks, Y. (2008) The Semantic Web and the Apotheosis of annotation. In Proc. IEEE Intelligent Systems.(May/June 2008)

Wilks, Y., Catizone, R., and Mival, O. (2008), The Companions paradigm as a method for eliciting and organising life data, In Proc. Workshop on Memories for Life, British Computer Society, London, March 18, 2008

Woods, W. (1980) Cascaded ATN grammars. *American Journal of Computational Linguistics*, 6(1):1—12.

Young, S. (2002). Talking to Machines (Statistically Speaking). Seventh International Conference on Spoken Language Processing , Denver, CO.

Young, S. (2008) Using POMDPs for Dialog Management, IEEE/ACL Workshop on Spoken Language Technology (SLT 2006), Aruba , 2006