A Robot in the Kitchen

Peter Wallis,
Department of Computer Science
The University of Sheffield
Sheffield, S1 4DP, UK
p.wallis@dcs.shef.ac.uk

Abstract

A technology demonstrator is one thing but having people use a technology is another, and the result reported here is that people often ignore our lovingly crafted handiwork. The SERA project - Social Engagement with Robots and Agents - was set up to look explicitly at what happens when a robot companion is put in someone's home. Even if things worked perfectly, there are times when a companion's human is simply not engaged. As a result we have separated our "dialog manager" into two parts: the dialog manager itself that determines what to say next, and an "interaction manager" that determines when to say it. This paper details the design of this SALT-E architecture.

1 Introduction

The SERA project, funded under FP7-ICT call 3, was initially intended to take established technology and put it in people's homes so we could record what happens. The core idea was to provide data in order to compare alternate methodologies for moving from raw data to the next generation of synthetic companion. Our primary motivation for the proposal was the realisation that the semantics of language is just one part of language in use. Even in apparently task based dialogs, effective repair strategies are essential and, what is more, highly dependent on social skills. Although there are many ways of looking at language, do any of them provide the kind of information, and level of detail, required to build better conversational agents?

The focus has turned out to be on robots rather than embodied conversational agents and the robot of choice was a Nabaztag. The Nabaztag is a commercially produced talking head from Violet in the style of Kismet and the Philips iCat. It is a stylized rabbit with expressive ears, a set of multi colour LEDs and is marketed as the world's first internet enabled talking rabbit. The rabbit connects to the Violet server via a wireless router and can run several applications including receiving SMS messages, weather reports, tai chi, and streaming selected radio or blog sites.

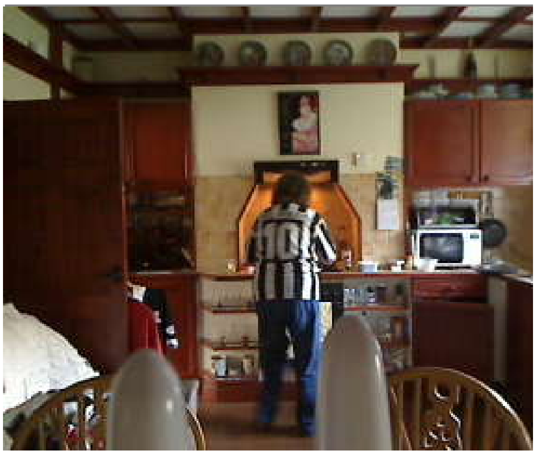

Figure 1: Making an omelette. In the real world, people ignore our handiwork! (note Nabaztag ears in the foreground)

The target participant group for the SERA experiments was older people with little experience of the limitations of computers. As it turns out, our subjects to date all have personal computers at home, but the lack of a keyboard or screen, and the rabbit being the only visible "beige-ware" means the set-up has been seen as sufficiently novel to provide classic discourse behaviour in spite of its limitations.

The original scenario was to have the rabbit provide classic internet services but our connection with the National Health Service (UK) through one of the participants provided impetus for us to use a health related theme and enabled us to recruit some interesting people through Help the

Proceedings of the 2010 Workshop on Companionable Dialogue Systems, ACL 2010, pages 25–30,
Uppsala, Sweden, 15 July 2010. ©2010 Association for Computational Linguistics

Aged (Hel, 2010), Aged Concern (Age, 2010) and similar organisations.

The primary result so far is that the established technology is seriously wanting. Our initial intention was to put a Nabaztag in people's homes pretty much as it comes out of the box. The problem is that these robots are intended to be entertaining rather than useful and the novelty soon wears off. As Mival et al point out (Mival et al., 2004) it is quite a challenge to design something that doesn't "end up in the back of the cupboard with the batteries out." Indeed these machines are expected to be on a desk, and to be poked and prodded to make them do things. For instance, the messaging function of the Nabaztag is certainly fun and useful, but there are two modes in its standard format: in the first the rabbit gives the message and assumes you are there. There is no sensing of the environment; the rabbit simply blurts it out. In the second mode it acts more like a classic answering machine and the user is expected to press a button to prompt a conversation about messages. Although this might be useful, it is acting exactly like a classic answering machine and we thought we could do significantly better by adding a PIR sensor - a standard home security passive infra red sensor that detects movement. We thus skipped the first version of our set-up and moved straight to a slightly more pro-active version that incorporated a PIR sensor to detect if the user was present. This is where the trouble starts, and is the primary point addressed in this paper.

The second piece of wanting technology is ASR — the automatic speech recognition. We initially considered a range of possibilities for the ASR and settled on Dragon Naturally Speaking, version 10 (DNS). In part this was driven by the fact that other projects were using it, and in part because of the DNS reputation. If we had gone for something else and it didn't work, well, people would have asked why we didn't use DNS. As it turned out, we could not get DNS to work with our set-up and for the first pass we resorted to yes/no buttons. Despite failing to get it working, using DNS was probably the right decision for exactly the reason given above. For the effort to have any impact however, other researchers need to know what happened and to this end the next section details our woes.

2 Speech Recognition

Speech recognition has been seen as "almost there" for twenty years and, from Furbys to interactive voice response phone systems, there are instances where the technology is useful. What is more, there is a body of work that points to the word recognition rates being less critical than one might assume (Wallis et al., 2001; Skantze, 2007). We allocated three months of a speech post-doc to get something working and expected it to take a week. We considered several options including DNS, Loquendo's VoxNauta which has a garbage model (see below) the Sphinx-4 system from CMU which is open source and in Java, the Juicer system (Moore et al., 2006) for which we have local expertise, and the ubiquitous HTK ToolKit which would certainly have the flexibility to do what we thought needed doing but would, no doubt, result in something cobbled together and unreliable. On the plus side we did have a single user that we could train but on the minus, we felt a head-set microphone was out of the question for the type of casual interaction we were expecting.

From the outset the intention was to use word spotting in continuous speech rather than attempting to parse the user's input. This was primarily motivated by the observation that successful NLP technologies such as chatbots and information extraction work that way. What is more, unlike dictating a letter or capturing an academic talk, we expected our subjects would not talk in full sentences, and utterances to be quite short. A command based system was considered but we did not want to restrict it to "Say yes, or no, now" style dialogs.

The approach we took was to use DNS as a large vocabulary continuous speech recognizer and then run regex style phrase spotting over the result - a classic pipeline model. The architecture was, and remains, an event driven model in which the dialog manager unloads and loads sets of "words of interest" into the recognizer at pretty well each turn. These sets are of phrases rather than words, and ideally would include the regex equivalent of ".+" and "^" - that is "anything said" and "nothing said". The recognizer then reports back whenever something of interest occurs in the input, and does it in a timely manner.

The motivation for integrating speech and language this closely is the belief that the dialog manager can have a quite concise view of what the subject will say next. What is more, getting it wrong is not critical if (and only if) the dialog manager has a decent repair strategy. The first of these beliefs is discussed further below, and the second is based on the results such as those in Wallis and in Skantze mentioned above.

The result was that we failed to get speech recognition working for the first iteration - despite the world leading expertise in the group. To quote from the 12 month project review:

> The COTS speech recognition did not prove as effective as supposed in the unstructured domestic environment, partly because of poor accuracy but also because of unacceptable latency imposed by the language model. Effective ASR deployment was further complicated by lack of access to the workings of the underlying proprietary recognition engine. ... and there is now a wider realisation and acceptance among partners that ASR is not a solved problem. [sera m12 review, 25/03/2010]

It turns out that a significant part of the performance delivered from dictation systems comes from the language model, not from the sound itself. The result was firstly that the system would wait for more input when the user didn't produce a grammatical sentence. This latency was often well beyond the point at which the resulting silence is treated by the user as information bearing. Secondly, when we did grab the available parts of the decision lattice in order to fix the latency issue, the hypotheses were very poor. Presumably this is because the language model was providing evidence based on the false assumption that the user would speak in proper sentences. Trials are under way to test this. The take away message is that dictation systems are not necessarily suited to interactive dialog. We have since heard that there are "secret switches" that those in the know can adjust (Hieronymus, 2009) on DNS but, in retrospect, if one is forced to use a COTS product one might be better off using a system such as VoxNauta that acknowledges the needs of interactive systems by including a garbage model. At least Loquendo have thought about the problems of interactive speech even if there is an apparent performance difference as measured in terms of word error rates.

The extent to which ASR relies on the language model encourages us further to believe that a tightly coupled dialog manager and speech recognition system will prove significantly better than simply piping data from one module to another.

3 Situated agents

If you use a chatbot, or trial a demo, you necessarily attend to the artifact. Your attention is on it, you want your attention on it, and the trial satisfactorily ends when you stop attending to it. Alarms are designed to demand attention, but what should a companion do? Figure 1 is a typical scene in participant number one's kitchen. She is making an omelette, and has told the rabbit that she is making an omelette. Now she is not attending to the rabbit and so what should the rabbit do? In particular, the rabbit can receive SMS style messages and if one arrives as she is making her omelette, should the rabbit pass it on now or wait until the next time she talks to it? There is of course no right answer to this but the issue does need to be managed. This is not a problem for a demo in which the action is scripted, and it is not an issue for the Nabaztag in its commercial form as it only knows when a message arrives, and when the user presses the button. With a PIR sensor however the system knows that someone is there, but are they paying attention? In the first iteration the system was cobbled together with a quite linear approach to system initiative. The latest version takes a slightly more sophisticated approach and distinguishes between three states at the top level. The system is:

- Sleeping – not seeing or hearing anything,
- Alert – "attending to" the person,
- Engaged - it is committed a conversation

The most obvious case of engagement is when the person and the machine are having a conversation - that is Listening and Talking to each other, however even if the conversation is finished, the system may still want to keep the context of the recent discussion. As an example the system might have finished its (system initiated) conversation about the day ahead and wait to see if the human wants to talk about their day before moving back to the Alert state in which the subject would need to go through the process of initiating a discussion.

These four states, Sleeping, Alert, Talking, or Listening (Engaged) are controlled by external

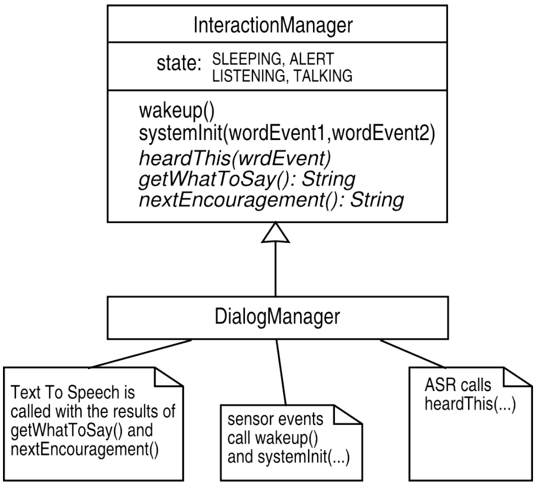
InteractionManager
state: SLEEPING, ALERT LISTENING, TALKING
wakeup()
systemInit(wordEvent1,wordEvent2)
heardThis(wrdEvent)
getWhatToSay(): String
nextEncouragement(): String
DialogManager
Text To Speech is called with the results of getWhatToSay() and nextEncouragement()
sensor events call wakeup() and systemInit(...)
ASR calls heardThis(...)

Figure 2: The InteractionManager handles *when* to say somethng; the DialogManager *what* to say.

events and timers. The GUI for editing dialog action frames provides 4 timing values as follows:

Pause 1	indicates the end of a turn by the user - it is an opportunity for the system to say something.
Pause 2	indicates the system ought to say something, and with nothing to say, it does an encouragement.
Pause 3	is the time after which the system drops the context of the conversation.
Pause 4	is the time at which the system goes to Sleep after the last PIR event.

Mapping these pauses into action, at pause 1 the system may move from Listening to Talking; pause 2 is the same but with a conversational "filler". At pause 3 it moves from Engaged to Alert, and pause 4 from Alert to Sleeping. The PIR sensor is the primary means by which the system is moved from Sleeping to Alert, and Alert to Engaged (actually Listening) can be human initiated by calling the system by name - "Hey Furby!" being used on that classic toy, and "Computer" being used on the bridge of the Star Ship Enterprise. Alternatively the system may initiate a conversation (Alert to Listening again) based on sensor information (for example, in our case the house keys being taken off the hook) an incoming message, or a diary event.

The SALT(E) interaction manager relates to the dialog manager in that the interaction manager handles the timing and determines when to say things while it is left to the dialog manager to decide what to say. The interface can again be described with a class diagram in which a DialogManager extends the InteractionManager implementing the following abstract methods:

```
heardThis(wrdEvent)
getWhatToSay():String
nextEncouragement():String
```

It is of course trivial to implement an Eliza style conversation based on heardThis/getWhatToSay with nextEncouragement taking the role of "nothing matched" patterns. In the case of SERA, the dialog manager is a conventional state based system with states clustered into topics.

The interaction manager also provides two other methods:

```
wakeup()
systemInit(WrdEvent1,wrdEvent2)
```

The first moves the system from Sleeping to Alert and initiates the pause 4 timer. The method systemInit(...) calls heardThis() immediately with wrdEvent1 - note the interaction manager still needs to call getWhatToSay() before anything is said. The second argument is past to heardThis() the next time the system becomes Alert. That is, the next time the user appears and the system moves from Sleeping to Alert, or the next time the system moves from Engaged to Alert. wrdEvent1 is an urgent message - in our case the message that the video recording is on - and wrdEvent2 represents something that can join the queue.

4 How language works (version 3)

The above has been rather low level but hopefully sufficiently brief, while detailed enough to be reproducible. But why is this of interest? Surely this is simply a technical issue that can be left to the RAs - a classic case of "flush pop-rivets" (Vincenti, 1990) which might be critical but is surely, well, boring. This section provides the theoretical background to the claim that managing engagement is critical.

The classic computer science view of human language is that it is some form of debased perfect language (Eco, 1995). In the middle ages perfection was defined in terms of God but to the Modern mind perfection has tended to mean something elegant, concise and unambiguous, typified by predicate calculus. Attempts to make computers understand language have forced the realisation that human languages are primarily driven by convention, highly context sensitive, and rely on the human capability for simile and metaphor. My latest view is that it is worse than that and that we pretty much make it up as we go along. This section briefly introduces a model of language from the Applied Linguistics community and shows how that model makes managing engagement critical.

In 2004 a group of us became interested in the way people tend to swear at conversational agents (de Angeli, 2005). In some work on an animal version of the Turing Test, there is some rather dramatic footage of a dog attacking an AIBO (Kubinyi et al., 2003). The interesting thing is that the dog warns the AIBO (twice) before throwing it across the room. The observation is that dogs, like people, are social animals and that the warning appears to be one mechanism for socialization of the young. When people abuse chat-bots, are they trying to socialize the machine? This of course would not be a concious process but rather normative (Wallis, 2005). This prompted a search for some high level social norm that might explain why people swear at computers. The result of that search was such a rule from the literature on Conversation Analysis or CA.

Paul Seedhouse (Seedhouse, 2004) summarises the outcome of the last 50 years of CA research as with the observation that a conversational partner's utterance will, in the normal case, go **seen but unnoticed**. That is, the utterance is an answer to a question, a response to a greeting or, more formally the second pair part of an adjacency pair. This is something computers can handle. A common occurrence is however where an utterance goes **noticed and accounted for**. In this case the conversational partner does not produce the expected response but what they do say makes sense. Indeed the conversational partner will work hard to produce the accounting. This is Eggins and Slade (Eggins and Slade, 1997) introducing sequential relevance:

> Consider, for example, the two turns at talk below:
>
> A What's that floating in the wine?
> B There aren't any other solutions.
>
> You will try very hard to find a way of interpreting B's turn as somehow an answer to A's question, even though there is no obvious link between them, apart from their appearance in sequence. Perhaps you will have decided that B took a common solution to a resistant wine cork and poked it through into the bottle, and it was floating in the wine. Whatever explanation you came up with, it is unlikely that you looked at the example and simply said "it doesn't make sense", so strong is the implication that adjacent turns relate to each other.

Having a machine account for human utterances is of course tricky but conversely the human conversational partner will work hard to account for what are actually machine failings.

If you fail to account for your CP's utterance, then your CP **risks sanction**. When your CP is a chat-bot, that sanction is likely, it seems, to take the form of swearing. As a classic example of a machine's interaction eliciting sanctions, here is a conversation from the 1996 Lobner Competition:

SYSTEM	G'day mate.
JUDGE	G'day. How's it going?
SYSTEM	My mother once said "How it is going is a mystery." She was a bit nuts.
JUDGE	Uh-huh
SYSTEM	Are you still learning to speak, my friend?
JUDGE	NO, no. But your anecdote took me a bit by surprise.
JUDGE	How was your morning?
SYSTEM	Oh well, that's a pity!
JUDGE	I'm sorry; what's a pity?
SYSTEM	Apologies accepted.
JUDGE	I feel compelled to insult you and we've only been talking a minute.
SYSTEM	We are bloody well talking, aren't we?
JUDGE	Oh baby. We sure are. So here are some keywords for you: shit mother baseball.
	...

This was the winning entry from that year but the point is not the system, but how the *human* responds.

There are of course more recent and better funded projects and in particular the DARPA Communicator Programme. The observation is that those systems did well when the human stuck to answering the system's questions and the ASR worked - the seen but unnoticed. When the grounded knowledge was not as the script designers expected however, the system did not have the world knowledge to understand the user's repair strategy. The systems also failed to hand over control to the user (Wallis, 2008). The result was sanction and although swearing is rare – surprising when one listens to the conversations – users did “not want to use the system on a regular basis” (Walker, 2002)

The mechanism for accounting for can be both tactical and strategic. Eliza and Parry were very successful in that user satisfaction was high compared to modern day systems. The mechanism was strategic in those systems in that they provide an accounting for their behaviour – in the first case because the role of psychologist accounts for the endless stream of personal questions, and in the second because being paranoid accounts for the system’s odd responses and interests.

4.1 So, engagement?

Why are we interested in engagement? Because in order for the human to “work very hard to find a way of interpreting [what the machine said]” the human must be committed to the conversation. This commitment needs management, and it is the role of the InteractionManager to do this. This is not an issue for a chat bot on a website nor for a system set up for experiments in a laboratory, but becomes a significant issue for an interactive artifact that is permanently in someone’s kitchen.

5 Conclusions

Our aim is to study long term relationships between people and robot companions and the intention is to put Nabaztags in an older person’s home and see what happens. This is not as straightforward as it may first appear as much of our understanding of these systems is based on demonstrators and experimental trials in which attention is, by the very nature of the trial, directed to the artifact. We introduce the SALT(E) model which separates the dialog manager in to a module that determines *what* to say, and another that determines *when* to say it.

6 Acknowledgments

The research leading to these results has received funding from the European Community’s Seventh Framework Programme [FP7/2007-2013] under grant agreement no. 231868. It has also received help from Loquendo that has generously provided the text-to-speech system for our robots.

References

2010. Aged Concern. http://www.ageconcern.org.uk.

Antonella de Angeli. 2005. Stupid computer! abuse and social identity. In Antonella De Angeli, Sheryl Brahnam, and Peter Wallis, editors, *Abuse: the darker side of Human-Computer Interaction (INTERACT ’05)*, Rome, September. http://www.agentabuse.org/.

Umberto Eco. 1995. *The Search for the Perfect Language (The Making of Europe)*. Blackwell Publishers, Oxford, UK.

Suzanne Eggins and Diana Slade. 1997. *Analysing Casual Conversation*. Cassell, Wellington House, 125 Strand, London.

2010. Help the Aged. http://www.helptheaged.org.uk.

Jim Hieronymus. 2009. personal communication.

Enikö Kubinyi, Ádám Miklósi, Frédéric Kaplan, Márta Gácsi, ózsef Topál, and Vilmos Csányi. 2003. Social behaviour of dogs encountering AIBO, an animal-like robot in a neutral and in a feeding situation. *Behavioural Proceses*, 65:231–239.

Oli Mival, S. Cringean, and D. Benyon. 2004. Personification technologies: Developing artificial companions for older people. In *CHI Fringe*, Austria.

Darren Moore, John Dines, Mathew Magimai Doss, Jithendra Vepa, Octavian Cheng, and Thomas Hain. 2006. Juicer: A weighted finite state transducer speech decoder. In *MLMI-06*, Washington DC.

Paul Seedhouse. 2004. *The Interactional Architecture of the Language Classroom: A Conversation Analysis Perspective*. Blackwell, September.

Gabriel Skantze. 2007. *Error Handling in Spoken Dialogue Systems - Managing Uncertainty, Grounding and Miscommunication*. Ph.D. thesis, Department of Speech, Music and Hearing, KTH.

Walter G. Vincenti. 1990. *What Engineers know and how they know it: analytical studies from aeronautical history*. The John Hopkins Press Ltd, London.

Marilyn et al Walker. 2002. DARPA communicator evaluation: Progress from 2000 to 2001. In *Proceedings of ICSLP 2002*, Denver, USA.

Peter Wallis, Helen Mitchard, Damian O’Dea, and Jyotsna Das. 2001. Dialogue modelling for a conversational agent. In Markus Stumptner, Dan Corbett, and Mike Brooks, editors, *AI2001: Advances in Artificial Intelligence, 14th Australian Joint Conference on Artificial Intelligence*, Adelaide, Australia. Springer (LNAI 2256).

Peter Wallis. 2005. Robust normative systems: What happens when a normative system fails? In Sheryl Brahnam Antonella De Angeli and Peter Wallis, editors, *Abuse: the darker side of human-computer interaction*, Rome, September.

Peter Wallis. 2008. Revisiting the DARPA communicator data using Conversation Analysis. *Interaction Studies*, 9(3), October.