University of Huddersfield

University of Huddersfield Repository

Cresswell, Stephen and Prigmore, Martyn

Feasibility report: Delivering case-study based learning using artificial intelligence and gaming technologies

Original Citation

Cresswell, Stephen and Prigmore, Martyn (2008) Feasibility report: Delivering case-study based learning using artificial intelligence and gaming technologies. Research Report. University of Huddersfield, Huddersfield. (Unpublished)

This version is available at http://eprints.hud.ac.uk/19906/

Feasibility report: Delivering case-study based learning using artificial intelligence and gaming technologies

Stephen Cresswell and Martyn Prigmore

Draft 0.99

Abstract

This document describes an investigation into the technical feasibility of a game to support learning based on case studies. Information systems students using the game will conduct fact-finding interviews with virtual characters. We survey relevant technologies in computational linguistics and games. We assess the applicability of the various approaches and propose an architecture for the game based on existing techniques. We propose a phased development plan for the development of the game.

Contents

1 Introduction

This report describes a technical feasibility study into a project to develop a computer game to support case-study based learning. A typical scenario is that the game characters play the roles of workers in a client business, and players take on the role of business systems analysts. The aim for the players is to elicit information about requirements from the game characters.

The intention is that the students will question the game characters by typing questions in English, and receive appropriate answers, which are extracted automatically from a corpus of case-study documents. To make the game more interesting and realistic, the characters should do more than just return factual answers: they should have plausible behaviours; they should have attitudes and opinions; some characters should be more helpful than others.

The vision of the game presents considerable technical challenges. Techniques from Artificial Intelligence and Computational Linguistics are needed to implement conversational agents, gaming technology is needed to represent the characters in graphics and audio, and considerable work is required from Information Systems staff to develop the case studies and characters in such a way that the case study scenarios are both interesting and cover pedagogically useful.

As a preliminary step to developing the game, this report surveys appropriate techniques and relevant existing systems which cover aspects of the requirement. The rest of the report is structured as follows:

Section 2 reviews the background of the use of case studies in teaching information systems. Section 3 reviews the requirements for game-based delivery of case studies. Section 4 reviews relevant systems and techniques from computational linguistics. Section 5 reviews the use of NLP in games. Section 6 considers the representation of conversational agents as virtual humans. Section 7 selects appropriate techniques given the features of the dialogue problem in the game. Section 8 proposes an architecture and development path for the game, and Section 9 concludes.

2 Background

Large case studies have previously been used in the school to teach undergraduate students in information systems. Students analyse a specific scenario from a real business, under the supervision of teachers. The case method (Lee 1987) provides an opportunity to put into practice concepts and techniques which have been taught in class.

To give the students practise at information gathering interviews, role-playing exercises were used, in which the teachers took the roles of employees in a client business, and the students would act as systems analysts.

The exercise was considered to be valuable for the students, but it was very time-consuming and repetitive for the staff, who were required to answer similar questions repeatedly from different groups of students.

General objectives of case method

- develop skills to investigate, analyse and plan,
- develop decision-making and problem solving skills,
- reinforce theory and techniques used in coursework
- develop organisation, teamwork and communication skills.

Skills put into practice from taught material are likely to include investigation, analysis, design (outline/detailed) and documentation.

2.1 Example: Hotel Ander Moselle

The Hotel Ander Moselle is a detailed scenario that has been used in the school as a case study. The scenario is described in (Lee 1987), but the teaching materials that were used in the school include more detail.

- Background briefing document
- Analysts briefing document (defining tasks set assigned to the analysts)
- Organisation chart, Building map
- Samples of forms (e.g. booking forms, invoices, ...), both blanks and completed forms.
- Briefing documents for worker roles. Each worker role is described on 1-2 typed pages. The description includes such details as:
  - duties
  - problems with current system
  - background information about person
  - attitudes, opinions, behaviour

  An example of a character profile is given below (describing the Hotel's chef):

  > "Pierre La Franc is an older man who has worked hard to achieve the position he now holds. He speaks very good English. He is very proud of his food and will talk endlessly and nervously about his excellent pastries, salads, etc. He is worried about the talk of expansion and is of the opinion that the analysts are checking how well he is doing his job. He has had a bad experience with Mr. Logan and is worried that the hotel is thinking of employing a graduate to do his job.
  > He is distressed about the stores aspect of his work; very often the stores are not sufficient. He sends his request to M. Logan and is not sure what happens to it. He tries to enforce his staff to write down what they withdraw but knows that they sometimes forget."

2.2 Example scenario: ServiceWatch

Computer-based systems for involving students in case studies have been previously developed within School of Computing and Engineering at University of Huddersfield.

The ServiceWatch system (Ward 1998) is a detailed hypertext-based representation of an equipment maintenance company, which was used to support development of systems analysis and modelling skills.

The initial pages present to the students an organisation chart, phone book, building map and marketing materials. Information gathering can be performed by following links to meet specific people, or to find examples of documents. The people can be interviewed by clicking on questions from a pre-defined set, and reading their responses.

(Ward 1998) acknowledges that browsing hypertext can be a passive, rather than active style of activity. In order to involve students in an active learning process, some separate goal must be given, In ServiceWatch, this is done providing activities separately.

The system was sold for use in around 40 other institutions. The success of ServiceWatch rests on the very detailed and realistic design of the scenarios.

3 Requirements

The proposed software is a game that simulates a client in a customer organisation by responding appropriately to fact-finding questions typed in ordinary English by the students. The system will be used in an information systems course as part of the case method of teaching.

The aim is to involve students in assimilating and analysing information presented to them, and to provoke them to formulate pertinent questions and consider the answers they obtains.

It is a requirement of the game to simulate sufficient aspects of the business to allow students to carry out a realistic requirements elicitation exercise with it.

It is *n*ot a requirement of the system to teach students subtle points of interview technique and interpersonal skills.

The system simulates characters in different job roles. The characters may vary in their knowledge, belief, attitudes, use of language and co-operativeness.

The characters are interviewed by the students through typed questions. Examples of questions [1] which the characters should be able to answer are:

- What is your job title?
- Who is your manager?

[1]Some taken from ServiceWatch

- What are your main responsibilities?
- How are the orders processed?
- What problems do you have with the current procedures?
- What effect does the delay have?
- What happens if the customer wants to change the quote details?
- For how long is a quotation valid?

Different students may be assigned to interview different characters. The students would then be expected to compare notes and carry out further fact-finding to resolve inconsistencies.

Use of game technology to embody characters with a visual manifestation will give characters a stronger distinguishing identity and allow body language signals to be given by the characters - e.g. they may become impatient, or defensive when questioned on a particular topic. Resolving conflicting accounts from different characters may be a part of the exercise.

The game will also simulate aspects of a business environment such as moving between offices and obtaining the opportunity to interview characters. Some of the sample documents will be obtained through the game (e.g. by requesting them from the game characters).

As far as possible, the game software should be easily re-configurable to different case scenarios. It should be possible to re-use existing scenario documentation as a basic set of responses for key questions.

3.1 Challenges

Developing the game will involve 3 main areas of work:

Design of the case scenario. The success of the case scenarios depends on a high level of detail, plausible characters and the availability of example data.

Artificial intelligence and natural language processing. This is a notoriously difficult problem, as simulated conversational agents will have obvious limitations.

Game graphics and speech Design of game graphics and sound is again very time-consuming and plays a key role in making the game engaging.

Whereas all three work areas are critical to the success of the project, and all three require considerable resources, we consider that the natural language processing aspect of the problem is the main area in which technical feasibility requires investigation.

4 Natural language systems

4.1 Chatbots

4.1.1 ELIZA

ELIZA (Weizenbaum 1966) is an early example of a conversation program, and is the ancestor of the general conversation programs that are today known as chatbots or chatterbots. ELIZA responds to English text input. An example conversation:

USER:	Men are alike
ELIZA:	IN WHAT WAY?
USER:	They are always bugging us about something or other
ELIZA:	CAN YOU THINK OF A SPECIFIC EXAMPLE?
USER:	Well, my boyfriend made me come here today.
ELIZA:	YOUR BOYFRIEND MADE YOU COME HERE?
USER:	He says I'm depressed much of the time.
ELIZA	I'M SORRY TO HEAR YOU ARE DEPRESSED.

The selfless contributions from ELIZA are mitigated by the scenario, that ELIZA is a non-directive psychotherapist. The patient (user) is prompted to keep talking openly, but the therapist does not direct the conversation on any particular topic. According to (Weizenbaum 1966): "This mode of conversation was chosen because the psychiatric interview is one of the few examples of categorised dyadic natural language communication in which one of the participating pair is free to assume the pose of knowing almost nothing of the real world."

Hence the goal of ELIZA is to generate any plausible response to keep the conversation going. There is no other communication goal. Although the illusion of understanding can be maintained for a few exchanges, it can soon become apparent that the responses are not based on understanding.

The responses are generated from a set of simple rules, triggered by key words and based on patterns. For example, a statement by the user "I am (X)" can be transformed to a reply "How long have you been (X)?"

The main lesson that can be drawn from ELIZA is that people are sometimes willing to attribute more intelligence to an artificial conversational agent than it really has. The idea of general conversation programs lives on. The annual Loebner contest is a form of Turing test in which conversation programs are tested and rated by human judges.

4.1.2 A.L.I.C.E.

A.L.I.C.E. is a general conversation program (winner of the Loebner prize in 2004) [2]. Its AIML representation (Wallace 2001) has been widely used for other chatbots, so seems a reasonable

[2] http://www.alicebot.org

```
<category>
<pattern>WHAT DOES THE WORD SHOE MEAN</pattern>
<template>A covering for the foot.</template>
</category>
<category>
<pattern>WHAT GOES WITHOUT SAYING</pattern>
<template>It is just an expression.</template>
</category>
<category>
<pattern>WHAT HAPPENED ON SEPTEMBER 11</pattern>
<template>The World Trade Center was destroyed</template>
</category>
<category>
<pattern>WHAT IS 1 PLUS 1</pattern>
<template>Two.</template>
</category>
```

Figure 1: Extract from an ALICE AIML file "general knowledge"

representative of current methods. The increased functionality of modern chatbots is achieved mainly by having very large sets of patterns, rather than any more sophisticated handling of language. AIML 'knowledge bases' are pattern databases. A example is shown in Fig. 1.

Reading this file shows a disappointing lack of abstraction. AIML does not specify much more than a mapping of input sentences to output sentences. There is some facility for rewriting patterns to other patterns, and setting values such as *topic* and *it*. The popularity of AIML may be due to its simplicity, which allows enthusiasts to write patterns easily, and the resulting AIML files can easily be shared and used together.

Whereas it may be possible to implement reasonably interesting behaviours with these mechanisms, they fail to capture the structure of the various kinds of knowledge. There is no separation of knowledge about words, grammar, discourse, and the domain.

The lack of structured knowledge in the implicit knowledge AIML 'knowledge base' is a problem. Many forms of knowledge are either not expressed or not expressed in an appropriate structure (see §4.2).

A more recent winner of the Loebner prize is Jabberwacky [3], which avoids the need for writing a large pattern database by simply collecting and re-using human responses from previous conversations.

4.2 Levels of representation

The chatbots discussed in §4.1 map from one surface representation to another. In this section we illustrate other levels of representation commonly used in computational linguistics.

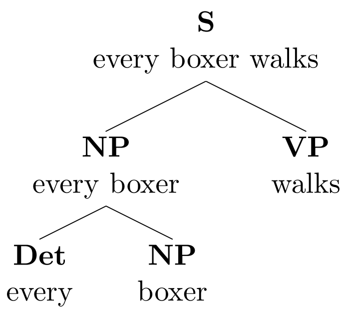
[3]http://www.jabberwacky.com

Syntax –

A language can be defined by a set of grammar rules. The simple example grammar and lexicon below (from (Blackburn & Bos 2004)) defines a language of sentences such as "Mia loves a boxer" or "Every woman likes a foot massage".

Sentence	s $\longrightarrow$ np vp
Nounphrase	n $\longrightarrow$ pn
	n $\longrightarrow$ det n
Proper name	pn $\longrightarrow$ [mia]
	pn $\longrightarrow$ [vincent]
Determiner	det $\longrightarrow$ [a]
	det $\longrightarrow$ [every]
Noun	noun $\longrightarrow$ [woman]
	noun $\longrightarrow$ [boxer]
	noun $\longrightarrow$ [foot,massage]
Verbphrase	vp $\longrightarrow$ iv
	vp $\longrightarrow$ tv np
Intransitive verb	iv $\longrightarrow$ [snorts]
	iv $\longrightarrow$ [walks]
Transitive verb	tv $\longrightarrow$ [loves]
	tv $\longrightarrow$ [likes]

Given such a sentence, a parser can be used to discover the syntactic structure of the sentence. E.g. for "Every boxer walks" we have:



Semantics –

Understanding a sentence can be considered as a process of converting the sentence into an appropriate expression in a Meaning Representation Language (MRL). The MRL is a suitable formal language chosen to facilitate inference and querying, hence first-order logic is a common choice. "Every boxer walks" could be represented in first order logic as $\forall x.(\text{BOXER}(x) \Rightarrow \text{WALK}(x))$

One approach to building up the meaning of a sentence is Montague Semantics (Dowty et al. 1981). Here, the principle of compositionality is used — the grammar is augmented with semantic composition rules which define how the semantic representation of each constituent is built from the semantic representations of the parts. Lambda calculus is used to specify substitution operations at each step. Augmenting our simple grammar gives the following rules:

s(NP@NP) $\longrightarrow$ np(NP) vp(VP)
n(PN) $\longrightarrow$ pn(PN)
n(DET@N) $\longrightarrow$ det(DET) n(N)
pn($\lambda p.p$@MIA $\longrightarrow$ [mia]
pn($\lambda p.p$@VINCENT $\longrightarrow$ [vincent]
det($\lambda u.\lambda v.\exists x.(u@x \wedge v@x)$) $\longrightarrow$ [a]
det($\lambda u.\lambda v.\forall x.(u@x \Rightarrow v@x)$) $\longrightarrow$ [every]
noun($\lambda(x)$WOMAN(x)) $\longrightarrow$ [woman]
noun($\lambda(x)$BOXER(x)) $\longrightarrow$ [boxer]
noun $\longrightarrow$ [foot,massage]
vp(TV@NP) $\longrightarrow$ tv(TV) np(NP)
iv($\lambda(x)$SNORTS(x)) $\longrightarrow$ [snorts]
iv($\lambda(x)$WALKS(x)) $\longrightarrow$ [walks]
tv $\longrightarrow$ [loves]
tv($\lambda x.\lambda y.(x@\lambda z.$LOVE$(y, z)$) $\longrightarrow$ [loves].
tv($\lambda x.\lambda y.(x@\lambda z.$LIKE$(y, z)$) $\longrightarrow$ [likes].

A parse tree with semantic expressions:

S
every boxer walks
$\forall x.(\text{BOXER}(x) \Rightarrow \text{WALK}(x))$

NP
every boxer
$\lambda v.\forall x.(\text{BOXER}(x) \Rightarrow v@x)$

VP
walks
$\lambda z.\text{WALK}(z)$

Det
every
$\lambda u.\lambda v.\forall x.(u@x \Rightarrow v@x)$

NP
boxer
$\lambda y.\text{BOXER}(y)$

However, even with these simple examples, there are difficulties with ambiguity. The sentence "Every boxer loves a woman" would be parsed by the grammar as:

$\forall b.(\text{BOXER}(b) \Rightarrow \exists w.(\text{WOMAN}(w) \wedge \text{LOVES}(b, w)))$

but in English, the following interpretation is also possible;

$\exists w.(\text{WOMAN}(w) \wedge \forall b.(\text{BOXER}(b) \Rightarrow \text{LOVES}(b, w)))$

Here, the scopes of the two quantifiers are exchanged, and the reading is that all boxers love the *same* woman. Handling such scope ambiguities has been the subject of much research. Most approaches involve producing an *underspecified* semantic representation, from which the alternative readings can be recovered.

Discourse representation A sentence is not always entirely self-contained – its meaning must be understood in the context of the preceding discourse, and in the context of the shared knowledge between speaker and hearer. A model of the discourse is necessary to solve problems such as:

- Resolving anaphoric pronouns (e.g. it, him). Such pronouns obviously refer to things or people who have recently been mentioned.
- Resolving ellipsis – i.e. where part of a sentence is missing, and is implicitly carried over from a previous utterance – e.g. consider the pair of questions:
  - "Where are the purchase orders filed?"
  - "What about the invoices?"

  The second question is lacking a verbphrase, but it is understood to mean "Where are the invoices filed?"
- Resolving presuppositions – pieces of information that are assumed in a context. For instance. For example, if we refer to "the wheels of a bicycle I borrowed", the use of the definite article to refer to the wheels suggests that they already exist as entities in the discourse, whereas in fact they are presumed to exist because of the newly-introduced bicycle.

Discourse Representation Theory (Kamp & Reyle 1993) addresses these problems by giving discourse a model-theoretic semantics, which can be built incrementally sentence-by-sentence. A good tutorial introduction is (Blackburn & Bos 2004)).

Dialogue – Within a dialogue, an utterance plays some role (e.g. a question or an instruction). Various approaches have been taken to modelling dialogue are considered in (§4.5).

Domain knowledge – Domain knowledge often takes the form of an ontology based on Description Logic (Baader et al. 2003).

Description Logics are a family of logics used for knowledge representation, which are based on restricting the expressiveness of first-order logic in order to make reasoning tractable.

A description logic ontology comprises: a set of classes, a set of subclass relations between classes ("isa"), a set of roles defining possible binary relations between individuals of classes (e.g. "responsible_for"), and a set of individuals (e.g. "Fred Schmit") and instances of binary relations between individuals (e.g. responsible_for("Ms Gardner","Fred Scbmit")).

The use of description logics by the semantic web community has led to the development of highly optimised DL reasoners such as RACER (Haarslev & Moller 2001). The careful compromise that has been made between the expressiveness of the logic and the computational tractability of the corresponding reasoning problems make these representations suitable for use in modelling a domain for a natural language application.

4.3 Open-domain question-answering

Recent surveys on question-answering systems can be found in (Harabagiu & Moldovan 2003), and (Bernardi & Webber 2007). The state-of-the-art in open-domain question-answering has been exhibited by the systems evaluated in the QA track of the annual TREC conference (Voorhees & Dang 2005).

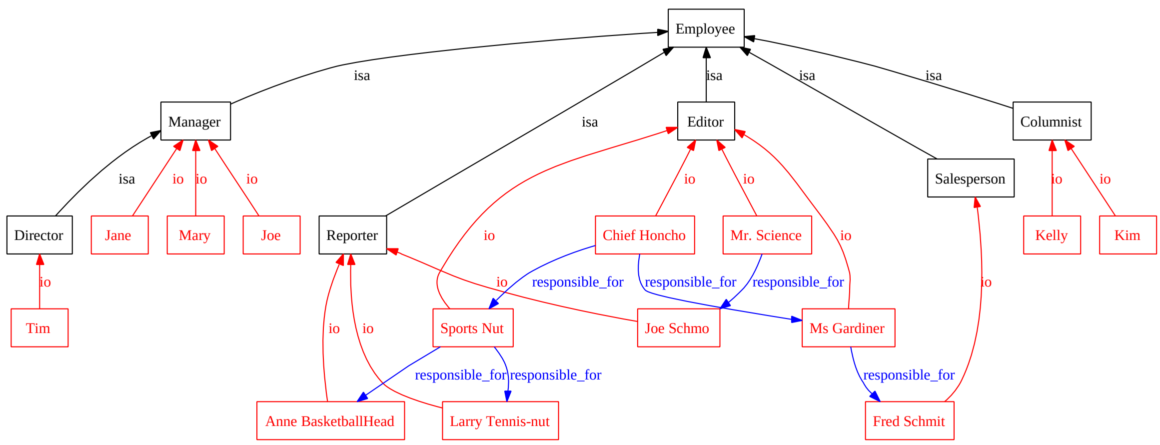


Figure 2: Part of an ontology describing a newspaper publishing organisation. The ontology is an example shipped with the Protégé-2000 ontology editor.

Open-domain question-answering usually uses a combination of information retrieval, extraction and statistical approaches. QA system based on document corpus (Hirschman & Gaizauskas 2001). The following sequence is typical:

1. Classify the question to determine the expected answer type.
2. Retrieve documents likely to contain answers to the query using important question words and related terms.
3. Analyse the retrieved passages to extract entities that are possible answers.
4. Rank the candidate answers to find the most likely answer.

The answers types in the TREC 2005 competition were restricted to persons, organisations, things and events. These are simple types of *factoid* answers.

4.3.1 Evaluation

For the proposed game, we would start with scenarios described by a relatively small corpus of documents. Hence we cannot rely on redundancy of representation to increase the probability of finding a fact stated in a way that the can easily be automatically extracted.

All of the information present in the documents must be extracted, even that which is implied. This means that deep, full text understanding of the text would be required in the style of (Balduccini & Baral 2007). More sophisticated analysis of the text brings with it a requirement for better grammar, lexicon, semantic analysis, a fuller domain model, discourse model and the capacity to infer facts which are not explicitly stated.

Hence, we judge that the difficulty of extracting answers from free text disproportionately high, given that there is a relatively small number of documents in each case scenario, and the burden of manually constructing a structured representation of the information they contain is acceptable.

4.4 Question-Answering from structured knowledge sources

(Molla & Vicedo 2007) gives a recent review of question-answering in restricted domains. (Frank et al. 2007) gives a detailed account of a system to answer questions in a particular domain, where information is represented in a variety of formats. Here, the role of a domain ontology is of increased importance. However, we should also consider systems in which factual information is represented directly in a database or knowledge base rather than in documents.

4.4.1 Natural Language Interfaces to Databases

A tutorial survey is found in (Androutsopoulos et al. 1995).

Development of natural language interfaces to databases (NLIDBs) is a mature area of work, dating back to work such as the LUNAR system (Woods 1973), which presented a natural language interface to a collection of data on lunar rocks.

For NLIDBs, the domain entities and relationships are already prescribed. Such systems precisely parse a natural language question into a logical query, search the database for the results, and (optionally) turn the answer back into natural language for output.

Hence, with relatively small, domain-specific lexicon and grammar, a system can translate natural language questions directly to queries in a meaning representation language.

An influential early example was CHAT-80 (Warren & Pereira 1982). Whereas this does not constitute the state-of-the-art, it is notable for its logical simplicity, availability and the transparency of its Prolog source code. The following are examples of English questions, intermediate logical representations of the questions, and answers produced.

- Does Afghanistan border China?
  - `ans(yes) <- borders(afghanistan,china)`
  - `yes`
- From what country does a river flow into the Persian Gulf?
  - `ans(C) <- river(R) & country(C) & flows(R,C,persian_gulf)`
  - `iraq`
- Which country bordering the Mediterranean borders a country that is bordered by a country whose population exceeds the population of India?

- ```
  ans(C) <- country(C) & borders(C,mediterranean) & country(C1) &
            country(C2) & population(C2,X) & population(india,Y) &
            exceeds(X,Y) & borders(C2,C1) & borders(C,C1)
  ```
- `turkey`

The queries that can be expressed in CHAT-80 are comparable to those that can be framed in SQL. Hence the English used there could be viewed as a restricted formal language. A regular user of such a system would develop an awareness of the limits of the language that is accepted by the system.

NLIDBs have been around for a long time, and commercial systems are available. Perhaps one reason why they have not been widely adopted is that a database schema itself does not contain any information about how to understand English sentences concerning the data. Work needs to be done to create a lexicon and grammar, or to train the system to understand relevant language.

A more recent view of current systems is in (Popescu et al. 2003).

4.4.2 Co-operative answering

Delivering raw facts in answer to a query is not close to the way that people behave when answering questions. People usually know or make assumptions about the state of knowledge of the other participant, and tailor their answers accordingly. (Grice 1975) introduced the Cooperative Principle, which states participants in a dialogue collaborate towards the common goals of the conversation. The cooperative principle gives a set of four maxims which are normally expected to be followed.

Maxim of Quantity
: Give the right amount of information. Make your contribution as informative as required, but no more informative.

Maxim of Quality
: Try to make your contribution one that is true. Do not say something that you believe to be false, or for which you lack sufficient evidence.

Maxim of Relation
: Be relevant

Maxim of Manner
: Avoid obscurity or ambiguity; be brief and orderly.

Research on Cooperative answers to questions deals with the problem that logically correct answers can often be seen as unhelpful or misleading, contravening Grice's maxims. For instance, if a user asks the question:

Are all drivers paid weekly?
  ```

If there are no drivers, the logic-derived answer to the question is 'yes'. However, a human answering the question would be likely to reply "There are no drivers". To produce such a response the answerer must identify that the question contains an incorrect assumption – i.e. that there are drivers. (Gaasterland et al. 1992) gives a survey of work in co-operative answering.

4.4.3 Evaluation

The bounded nature of the domain in our problem is an indicator that this form of deep processing can be applied. We can build a domain ontology from which factual questions can be answered. We can build a lexicon and grammar to handle questions about the concepts in the domain.

However, factual queries returning factual answers do not constitute a conversation. The game's conversational agents should not simply return the factual answer, but make use of it the factual answer in its response.

Another limitation of such an approach is that the understanding of language input is brittle, in that if a system fails to parse the English query, it is unable to do anything. This is appropriate for querying databases, where it is helpful to let the user know that input was not parsed. For a conversational agent, it is more appropriate to give a conversational response, even if it is based on only a partial comprehension of the user's utterance.

4.5 Dialogue models

A very simple model of a dialogue is a finite state machine (Fig. 3). This models a simple task-oriented dialogue in which the computer has the initiative to direct the flow of the dialogue, but the human responses determine the specific path through the network. Note that in the described example, the user responses are very specific and short (e.g. yes/no/other), but in general there may be any amount of analysis on user input, as long as this results in a categorisation into a predefined set of outcomes.

Whereas this model of dialogue would be sufficient for some purposes such as carrying out simple formulaic transactions, it is not a very general model of dialogue, and will not be sufficient for human-computer interactions in which the human has the initiative.

A number of tutorial dialogue systems have been developed in which the simulated tutor presents the student with a scenario or a problem (e.g. a physics problem) and the student has the task of producing an explanation (Graesser, VanLehn, Rose, Jordan & Harter 2001). The required explanation typically comprises a sequence of reasoning steps.

The form of dialogue modelling used in these systems needs to be a little more sophisticated.

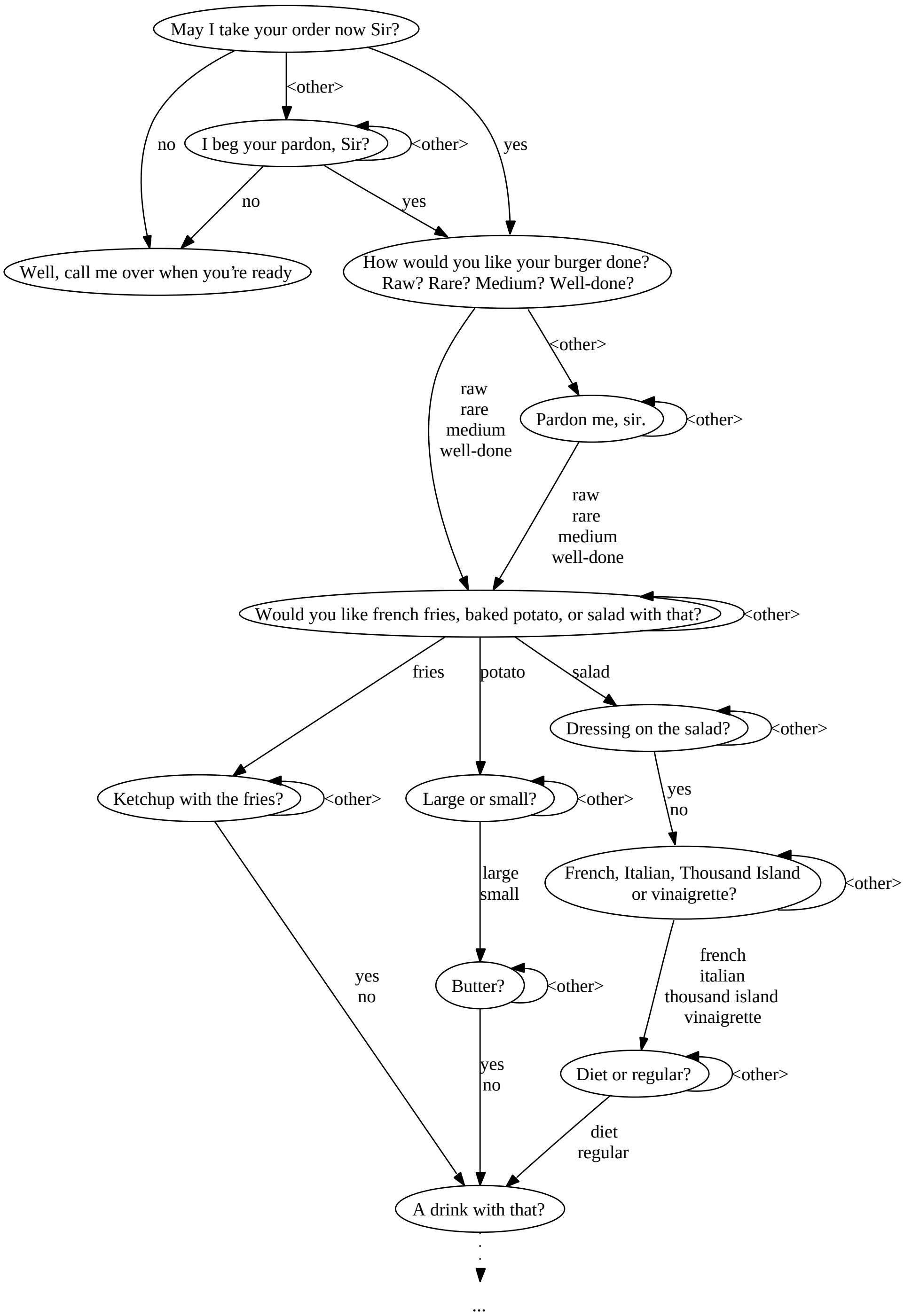


Figure 3: Finite state machine representation of a dialogue. Based an a dialogue from the game “Bureaucracy”, whose main purpose is to frustrate the user.

4.5.1 Tutorial dialogue systems

In tutorial dialogue systems, the computer usually plays the role of tutor, and usually controls the flow of dialogue by asking questions.

(Burton & Brown 1979) describes the NLP component of SOPHIE, an early system intended to teach troubleshooting of electronic circuits. The student is required to ask questions about the circuit (e.g. request voltage measurements) in order to diagnose a fault. SOPHIE uses a mathematical model of the circuit to answer questions about its behaviour. The language understanding component used a *semantic grammar*. This is a grammar in which semantically meaningful entities in the domain are used directly as syntactic categories. This is a way of specialising a grammar for a limited domain. This allows the grammar to be more efficient than more generally-applicable syntax rules. The disadvantage of this approach is that more work is required to adapt the grammar to another domain. However, semantic grammars are widely used in practical NLP systems.

4.5.2 Autotutor

Autotutor is an example of a tutorial dialogue system where a student is guided to explain a scenario

The student has the option to take the initiative in the dialogue by asking questions. The simulated tutor has a strategy to nudge the student through their explanation by dropping increasingly heavy hints. At each dialogue turn, the student may respond as prompted by the tutor, or may ask a question

Tutor:	When you turn on the computer, how is the operating system first activated and loaded into the computer.
Student:	When the computer is turned on a file is automatically booted up.
Tutor:	Anything else?
Student:	The file contains the operating system.
Tutor:	What else?
Student:	The operating system is then read into RAM and activated.
Tutor:	Right. The CPU executes instructions in ROM, which helps the operating system to start.
Student:	After being read from ROM the OS is activated.
Tutor:	Right, let's review what we've just gone over ...

Student utterances are categorised as:

- Assertion
- WH-question
- Y/N-questions

- Metacognitive comments (e.g. I don't understand).

Latent semantic analysis (LSA) is used to match student responses with prototype good and bad responses. LSA is based on statistical match between sets of words in student utterance and prototypes to measure how semantically close they are.

The benefits of such an approach are its robustness and the simplicity with which new dialogue scenarios can be constructed.

However, LSA treats a sentence as being a bag of words, and does not analyse its structure, so cannot be used to obtain a deep understanding of the student utterance.

Autotutor uses a finite state machine within a single dialogue turn to categorise the user's utterance and to provide a response, often in several stages. A multiple-part response may comprise, an initial response to the student's input (e.g. an answer to a question or an acknowledgement of statement with positive, negative or neutral feedback, together with a further prompt to move the student's reasoning e.g. a hint at part of solution that is missing, and a clear cue that the student is expected to contribute a further explanatory sentence.

4.5.3 Other dialogue models

According to (Jurafsky & Martin 2000, Ch. 19) Dialogue is characterised by

Turn-taking with adjacency pairs, in which a speaker hands over to their conversational partner, with an expectation of a particular type of response. Examples are pairs, such as greeting–greeting, question–answer, request–grant.

Grounding , in which participants reflect their common ground by acknowledging their level of understanding and acceptance of what has been said.

Implicature , which concerns the understanding of the significance utterances beyond their literal meaning

(Austin 1962) considered all utterances to be a kind of action (speech act), and that this can be seen at three levels:

Locutionary act - the act of speaking a sequence of words.

Illocutionary act - the act of asking, requesting, informing, denying, etc.

Perlocutionary act - the actual effect on the thoughts and actions of the hearer.

For example, locutionary act of saying “Can you pass the salt?” is likely to correspond to an illocutionary act of requesting the hearer to pass the salt, rather than a request for the hearer to provide information about her ability to pass the salt. The problem from a computational

Forwards dialogue acts:

- Statement
  - Assert
  - Reassert
  - other-statement
- Influencing Addressee Future Action
  - Open-option
  - Directive:
    - Info-Request
    - Action-Directive
- Committing Speaker Future Action
  - Offer
  - Commit
- Performative
- Other Forward Function

Backwards dialoge acts:

- Agreement
  - Accept
  - Accept-Part
  - Maybe
  - Reject-Part
  - Reject
  - Hold
- Understanding
  - Signal-Non-Understanding
  - Signal-Understanding
    - Acknowledge
    - Repeat-Rephrase
    - Completion
  - Correct-Mis-speaking
- Answer
- Information-Relation

Figure 4: DAMSL categorisation of dialogue acts

viewpoint is that identifying the illocutionary act requires an understanding of the speaker's *intention*.

In the literature on computer dialogue systems, the term used is often *dialogue act* or *dialogue move*. The DAMSL initiative was a project to develop a common taxonomy of dialogue acts (Core & Allen 1997). Examples of their categorisations given in Fig. 4. The forwards dialogue acts represent an initiative by the speaker, whereas the backwards dialogue acts represent a reponse to the previous utterance.

(Searle 1975) considered indirect speech acts, in which the surface meaning of an utterance such as "Can you ...?", appears to be a yes/no-question, but is often a request for action. Here, utterances are understood as actions requiring the hearer to infer the intention behind them.

(Cohen & Perrault 1979) introduced an AI planning model for generation of speech acts. (Allen & Perrault 1980) applied planning to to comprehension of indirect speech acts, characterising it as a problem of *plan recognition*. The analogy to AI planning is that the hearer perceives the speaker's utterance as a step in a plan and infers the goal for which this step would be required, This general approach is known as the beliefs, desires, intentions (BDI) model.

4.5.4 Planning-based dialogue model

TRAINS-95 example (Ferguson et al. 1996)

The domain is planning rail transportation of freight. The user and computer collaborate in a task. The dialogue manager models the state of the dialogue, its own intentions and the user's requests, goals and beliefs.

Plan recognition is used to understand user's goals. Planning is used to select an appropriate dialogue move, and planning at the domain level is used to find potential solutions to the transportation problem.

M: I need to get to Detroit, Syracuse, and Montreal.
M states goal - but this is really a request for S to formulate a plan
S: OK *[highlights three cities]*
M: Let's send the trains from Chicago to Detroit.
(S recognises that M is forming plan to meet goal of moving train to Detroit. S informs M of problem with plan)
S: OK. *[draws route from Chicago to Detroit via Toledo]*
But the terminal at Toledo is delaying traffic due to bad weather. Trains take an additional five hours to move through there. *[highlights Toledo in red]*
M: That's OK. Now let's go from Atlanta up to Syracuse.
(S recognises that M has moved on to next goal - moving train to Syracuse. S requests more information.)
S: Please tell me a route to use to get from Atlanta to Syracuse
[Highlight engine Atlanta and Syracuse]
M Go via Charleston and Pittsburgh
(S forms plan using provided information, but detects further problem)
S: OK. *draws route via Cincinnati*
But traffic through Cincinnati is being delayed due to construction.
M: OK, let's go via Charleston and Buffalo instead.
(S recognises that a new plan is formulated)
S: No problem. *[draws route]*

So here we see a sophisticated dialogue, in which the system is capable of recognising the goals and plans of the human. That includes both the plans about the domain (which trains to move where) and about communication e.g. what to say next, and how to say it.

This dialogue is reported in (Ferguson et al. 1996) and also considered in (Haller 1999).

4.5.5 Information state dialogue model

The work of (Larsson & Traum 2000) describes the TRINDI architecture, a general model for representing dialogue, which subsumes simpler approaches. It is based on the *information state* model of dialogue. The information state refers to the mental state of each participant in the dialogue.

The information state of a participant may comprise the knowledge they had already at the start of the dialogue, the knowledge they acquire during the dialogue, and their current motivations to action.

Fundamental to the model is the idea of information state update. Updates are typically related to the observation and performance of dialogue moves. For example, statements add propositional information, and questions provide motivation for others to provide specific statements.

Information state theory of dialogue consists of:

informational components - e.g. the state of knowledge, goals etc. of dialogue participants.

formal representations of the informational components - i.e. datastructures describing the informational components.

dialogue moves that trigger the update of the information components.

update rules : declarative rules to select a dialogue move given current information state.

update strategy : select most appropriate rule where > 1 rule applies

The TRINDI approach gives a formalisation of information state, which allows specific theories of dialogue to be built and tested. The information state approach subsumes both finite-state and plan-based dialogue models, allowing features of both to be combined.

TrindiKit is a software package for constucting dialogue managers based on the information state model. This has been used to construct various dialogue systems.

4.5.6 Evaluation

Models of dialogue based on planning and plan recognition are an impressive demonstration of non-trivial dialogue systems, which *know what they are talking about.* We judge that these models are more sophisticated than are required for the game. However, it is valuable to identify dialogue modelling as separate activity. Our system should at least classify different types of appropriate move, and respond to them according to some model that will tailored to our specific requirements.

4.6 Natural Language Generation

An overview of language generation is given in (Bateman & Zock 2003). The following account is paraphrased from (Jurafsky & Martin 2000).

A simplistic approach is to use canned text or a template-based approach.

Content selection – What information is needed to achieve the communication goal?

Lexical selection – Select lexical item most appropriate for expressing textual items.

Sentence structure **Aggregation** – Allocate the selected content into phrase and sentence chunks.

Referring expressions – Decide how to refer to domain objects.

Discourse structure – Muti-sentence discourse must have a clear, coherent structure.

A typical architecture for generation consists of two stages:

Discourse Planner – which must resolve all of the above choices.

Surface Realiser – which takes a fully specified discourse plan and generates individual sentences according its grammatical and lexical rules.

Generating textual output from a meaning representation language can be considered as the inverse of the parsing problem. Unfortunately, it is generally not possible to make use of the same grammar formalism for both parsing and generation, as both place different restrictions on . A widely-adopted grammar formalism used for generation has been Functional Unification Grammar (Kay 1980), which is the basis of the FUF text generation package (Elhadad 1993).

Research on generating appropriate answers in the domain of intelligent help systems for UNIX users in described in (Wilensky et al. 2001), (Chin 2000). The emphasis is on tailoring the answers appropriately to the user's level of knowledge.

We will adopt a simple approach of template filling, together with canned text. Text generation is a complex area in its own right.

4.7 Natural language tools

Whereas there are many small demonstrator systems available that provide the machinery for constructing a dialogue system from the ground up (e.g. software accompanying (Gazdar & Mellish 1989), (Blackburn & Bos 2005), (Warren & Pereira 1982)) this would require the

1. Adaptation of existing code for parsing, semantic interpretation, discourse processing.
2. Construction of domain-specific grammar and lexicon.

The benefit of building code in this way is that it is possible to implementor to understand and adapt the code at every level. However, it may be impractical to scale these systems to cope with lexicons beyond a few hundred words. Also the parsers mentioned are presented for instructional purposes, and are unlikely to scale well to large grammars.

An alternative approach is use an already-developed parser with a wide coverage of English (e.g. Heart of Gold (Schäfer 2006)). This has the advantage of robustness, in that structures are derivable for a high proportion of input utterances that are in well-formed English. The semantic representation is based on Minimal Recursion Semantics (MRS) (Copestake et al. 2005). As there is often an ambiguity to the structure of English sentences, in general multiple alternative structures may be returned by parsers and semantic analysers. A current approach to dealing with this problem is to return a single structure, which is itself ambiguous, together with a set of constraints on how the ambiguity can be resolved. An unfortunate side-effect of taking such an approach is that the representations are complex and difficult to interpret.

5 NLP in games

In the early 1980s, adventure games were a popular format. These were modelled after the "Colossal Caves" game by Will Crowther and Don Woods. The early games used text-only games, in which the player typed English commands at the keyboard, and the machine produced a text description of the player's current situation (e.g. effects of last command, description of location). Although the language handling in the early games was quite simple, usually consisting of verb-noun pairs such as "take gold" or "throw axe", they did at least have the property that it was not possible to see in advance all of the actions that were possible, and players had to think up plausible actions and experiment to find out what would work.

Some games have been written in the detective genre, and these could be considered the closest in spirit to our system analysis game.

As graphics capabilities became more sophisticated, adventure games became a mixture of text and graphics, and eventually the textual aspect has largely died out. Today's equivalent impressive 3D scenes, moving, speaking, 3D characters, sound effects and music. Unfortunately, controlling a game by free text input is no longer used. Interaction is typically by a game controller, and the repertoire of available actions is fixed and always apparent. So as games have become more sophisticated, the most interesting mode of interaction - the one that made games feel open-ended, has been lost.

In contemporary games, dialogue with game characters is typically done by selecting a canned utterance from a menu, hence it is not possible for a player to freely compose their own utterances.

Recent work to use more sophisticated language processing can be found in (Kacmarcik 2006), (Kacmarcik 2005) which propose ways of constructing dialogues with non-player characters. However, the mode of use is still via menus.

(Koller et al. 2004) describes a recent implementation of an adventure game using state-of-the-art NLP techniques to make the language handling more sophisticated.

6 Embodied conversational agents

It is part of the requirement to render characters graphically. We want the characters to speak their responses. In addition to the the benefit of making the game more engaging, it opens up the possibiity to include some non-verbal communication, indicating that the characters are impatient, bored, nervous, lying etc. In addition to textual output, embodied agents have other modes of expression available:

- Prosody – i.e. *how* words are said (intonation, speech rate, intensity)
- Facial expression
- Gaze
- Gesture

(Johnson et al. 2000) reviews a number of applications using embodied agents for teaching.

However, tbe design decisions about the form of output have an influence on how the responses are generated. More realistic speech requires that sounds are pre-recorded, but this means that all responses must be “canned” in advance. Synthesised speech is not as naturalistic, but would allow more flexibility in responses. Whereas it is possible to generate speech with reasonably natural prosody (Dutoit & Styliannu 2003), marking up speech output with additional annotations is necessary if the communication is to carry any information that was not implicit in the text.

(Pelachaud & Bilvi 2003) describes a system (*Greta*), in which multimodal communcations are described using APML (affective presentation markup language). Utterances are annotated with communicative functions, which include belief state (e.g. certainty or uncertainty), intention (e.g. inform, request, cue turn-taking) affective state (i.e. their emotions), meta-cognitve state (e.g. breaking gaze while thinking). An example is shown below.

```
<APML>
<turn-allocation type="take turn">
<performative type="greet">
Good Morning, Angela.
</turn-allocation>
<affective type="happy">
It is so <topic-comment type="comment">wonderful</topic-ocomment> to see you again.
</affective>
<certainty type="certain"> I was sure we would do so, one day! </certainty>
</APML>
```

A text-to-speech package is used to convert the text into a sequence of phonemes, along with their durations. The phonemes are then used to generate *visemes*, which describe lip movement.

The APML is used to annotate the text with information about communicative function. A conversion process maps cummunicative functions into facial signals, which are synchonised with the phonemes.

Both the facial signals and the visemes are then encoded as MPEG-4 facial action parameters (FAPs), which describe individual movements of facial features that can be rendered graphically.

Numerous other markup languages for agent expression and behaviour have been proposed. (Kopp et al. 2006) gives a review and describes an intiative by several groups of researchers to define a standard. Multimodal generation can be seen as an extension of text generation. As such, approaches used are often based on planning.

7 Assessment of approaches

7.1 Question-answering from documents

Extracting answers from documents is not considered feasible (see §4.3.1), However, machine readable documents could be used in an interactive process to aid the construction of knowledge sources to represent scenarios for the game. For instance, non-dictionary words present in the source documents are likely to be domain-specific terminology, which will need to be represented in a domain ontology. Additionally, it may be appropriate to select some sections of source text to be used directly as canned dialogue responses.

Conducting Wizard-of-Oz trials (i.e. using human-human dialogue, but communicating only via typed messages) would allow the accumulation of a corpus that could be subsequently used for automated question-answering.

7.2 Distinguishing features of our dialogue task

Domain model The domain is limited. It is possible to build an ontology of the domain. The model generally remains static – it is not necessary to update information about the world state as a result of understanding a student utterance. However, it will be necessary to update the character's knowledge state about the what the student has already been told.

User model. The main aspect of the user that we wish to model is which of the main points have been explained.

Initiative. The human usually has the initiative, but the game characters should do more than passively answer questions. They should make remarks, offer opinions that may influence the flow of conversation, and may sometimes seize the initiative (e.g. to enter a set-piece sub-dialogue).

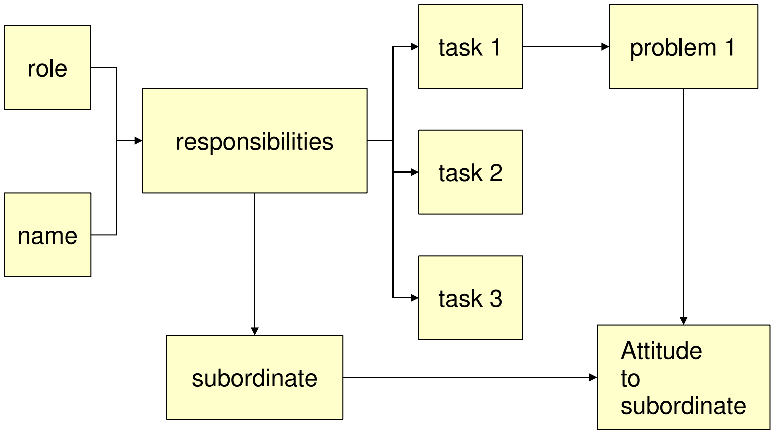


Figure 5: Precedence relations between topics

7.3 Dialogue requirements

It is possble to model the key points to be covered by the character and dependencies between them. For example, topic C can't explained unless topics A and B have been explained, e.g. Fig. 5. The user model keeps track of which of the main points have been explained.

There are order dependencies between points, but the order cannot simply be fixed, as the human player has the initiative. The game character can, after giving the answer to a question, add a supplementary comment which hints at the areas that remain to be covered. This would allow the character to influence the flow of the conversation. Alternatively, an unhelpful character may attempt to steer dialogue towards an irrelevant topic.[4]

Additionally, we may have "set piece" sub-dialogues, in which the game character takes the initiative.

8 Proposal

8.1 Proposed architecture

Fig. 6 shows a sketch architecture for the propsed dialogue sysetem. The **parser and semantic analyser** make use of the lexicon and grammar to derive a semantic representation of the player utterance. Information specific to the domain is present in the grammar and lexicon, and the parser has access to the domain model to help resolve ambiguities. The output of this stage is one or more semantic representations of the utterance. It is possible that the utterance cannot be fully parsed. In the case the output will partial parsing information, e.g. only identifying that the sentence is wh-question and contains particular nounphrases.

A step to **resolve discourse entities** identifies nounphrases in the utterance (e.g. "him",

[4]It is at this level that M. La Franc's enthusiasm for pastries can be represented.

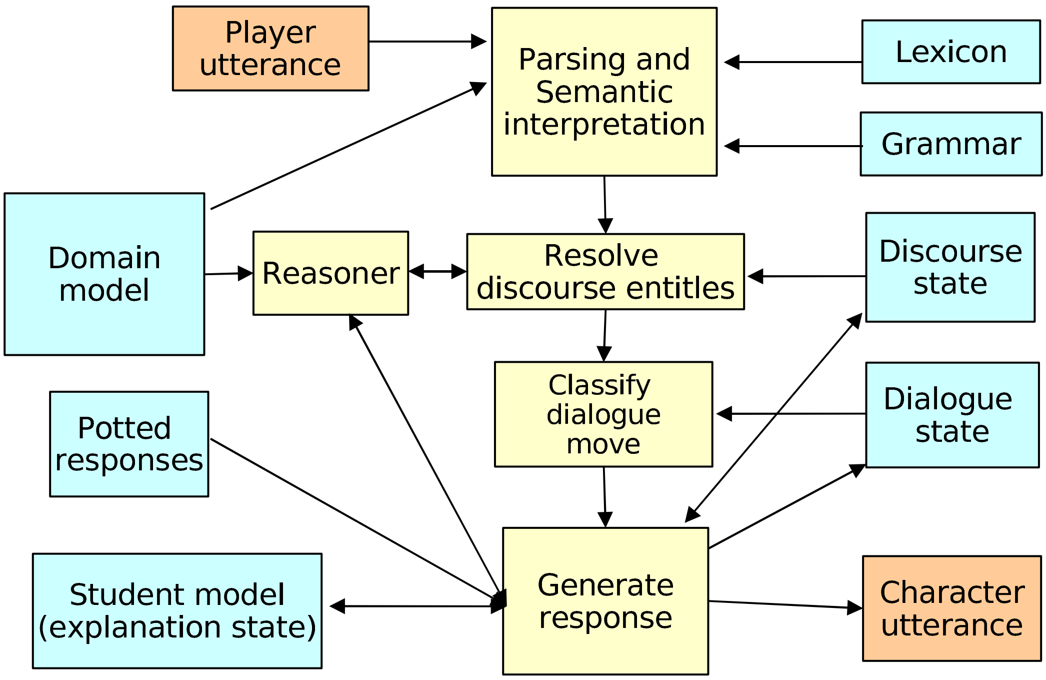


Figure 6: Proposed architecture

"your boss", "Bob") with individuals in the ontology. This requires access to the domain, and this should be mediated through a reasoner to process queries and make inferences.

The next step is to **classify the dialogue move**, and this depends partly on the current *dialogue state*, e.g. are we expecting an answer to our last utterance? Classification of the dialogue move determines the appropriate strategy for **response generation**. If the utterance is a fully understood factual question, then the appropriate response is to look up the answer in the knowledge base, generate the text for the answer (e.g. using template associated with knowledge base facts), and then return the answer along with a linking comment. Some question types will trigger a longer canned explanation of a particular point, and the model of the **explanation state** must be updated to record that this point has been covered. Incompletely parsed utterances trigger shallow responses on the basis of the partial parse information, e.g. that a particular person has been mentioned.

8.2 Development plan

The architecture described in §8.1 describes a natural language dialogue system. This itself requires work to

- Select a suitable part of a case study (e.g. around a single character).
- Develop domain model, lexicon, grammar (including semantic rules), dialogue moves, explanation structure, templated and canned responses.
- Iteratively develop and evaluate until reasonable coverage is achieved

It may also be worth exploring the use of weaker but more robust techniques based on information extraction and statistical matching techniques such as latent semantic analysis. It is possible to layer a deep-but-brittle approach with a shallow-but-robust approach as in (Aleven et al. 2001*b*). Any of the above approaches would benefit from obtaining dialogue transcripts through Wizard-of-Oz studies, in which the role of the game character is replaced by a human, but communication is still through the computer.

The other major challenges of developing the game, such as the development of interesting and detailed case material, and the development of the multimedia aspects of the game, could be investigated in parallel with the above work.

More consideration should also be given to the modelling of emotional state in the game characters. We have given some consideration to the expression, but not the generation of emotion in the characters. Plausible progressions of the character's emotional state would obviously make the game more engaging.

We have not considered the use of speech recognition in the game, although some of the dialogue systems that we have mentioned actually do make use of speech recognition. We feel that speech recognition increases the difficulty of a problem which is already very difficult. Whereas the technlogy is already available off-the-shelf, the problem is that it would reduce the quality of the human input to dialogue system. This is partly because recognition errors are unavoidable, and also because by spoken dialogue is often less tidy than written dialogue - e.g. by the presence of "um" and "er", pauses and partially complete sentences.

9 Conclusions

We have considered the requirements of the proposed game. The main technical problem is that of natural language dialogue.

We have surveyed existing techniques and systems covering aspects of the problem. Although natural language dialogue is a very challenging problem, to which a perfect solution is not practical, there are many examples of computer dialogue systems working successfully in limited domains. We have proposed a solution to the problem of implementing character dialogue, which is essentially a combination established techniques.

The original aim to extract answers directly from documents is considered very challenging, given the small size of the corpus of documents supplied, and the lack redundancy therein. However, this aspect is not considered necessary to the success of the project. A manual preparation of domain knowledge could be used in a pilot project, and the question of automatic extraction could be reconsidered at a later date. One possible approach is acquire a large corpus by recording human-human dialogues mediated through a computer.

However it is inevitable that some proportion of input utterances will not be properly understood by the natural language understander. Development must involve much experimentation, and the coverage of the natural language would develop incrementally.

10 Acknowledgements

Thanks to Steve Wade, Sam Clayton, Daryl Marples, Robert Ward, Barry Lee, Dave Brignell, Lee McCluskey, and Rob Lloyd-Owen for contributing help, ideas and documents.

References

Aleven, V. A., Popescu, O. & Koedinger, K. (2001*a*), A tutorial dialogue system with knowledge-based understanding and classification of student explanation, *in* 'Second IJ-CAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems'.

Aleven, V., Popescu, O. & Koedinger, K. R. (2001*b*), Pedagogical content knowledge in a tutorial dialogue system to support self-explanation, *in* 'AIED workshop on tutorial dialogue systems'.

Allen, J. (1995), *Natural language understanding*, 2 edn, Benjamin/Cummings.

Allen, J. & Perrault, R. (1980), 'Analyzing Intention in Utterances', *Artificial Intelligence* **15**, 143–178. Reprinted in (Grosz et al. 1986), pp. 441–458.

Androutsopoulos, I., Ritchie, G. D. & Thanisch, P. (1995), 'Natural Language Interfaces to Databases – An Introduction', *Natural Language Engineering* **1**(1), 29–81.

Austin, J. L. (1962), *How to do things with words*, Harvard University Press. Partially reprinted in (Kasher 1998), vol. 2, pp. 7–26.

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D. & Patel-Schneider, P. F. (2003), *The Description Logic Handbook: Theory, Implementation, and Applications*, Cambridge University Press.

Balduccini, M. & Baral, C. (2007), Knowledge Representation and Question-Answering, *in* 'Handbook of Knowledge Representation', Elsevier.

Bateman, J. & Zock, M. (2003), Natural Language Generation, in *The Oxford Handbook of Computational Linguistics* (Mitkov 2003), pp. 284–304.

Bernardi, R. & Webber, B. (2007), 'Question and Answers: theoretical and applied perspectives (editorial)', *Journal Of Applied Logic* **5**, 1–2.

Blackburn, P. & Bos, J. (2004), *Working with discourse representation theory: an advanced course in computational semantics*.

Blackburn, P. & Bos, J. (2005), *Representation and inference for Natural Language: a first course in computational semantics*, CLSI publications.

Bui, T. H. (2006), Multimodal dialogue management - state of the art, Technical Report TR-CTIT-06-01, Centre for Telematics and Information Technology, University of Twente.

Burton, R. R. & Brown, J. S. (1979), 'Toward a natural-language capability for computer-assisted instruction', pp. 605–625. Reprinted in (Grosz et al. 1986) pp. 605–625.

Chin, D. (2000), 'Strategies for expressing concise, helpful answers', *Artificial Intelligence Review* **14**, 333–350.

Cohen, P. R. & Perrault, R. (1979), 'A Plan-Based Theory of Speech Acts', *Journal of Cognitive Science* **3**(3), 177–212.

Copestake, A., Flickinger, D., Pollard, C. & Sag, I. A. (2005), 'Minimal Recursion Semantics: An Introduction', *Research on Language and Computation* **3**, 281–332.

Core, M. G. & Allen, J. F. (1997), Coding Dialogues with the DAMSL Annotation Scheme, *in* D. Traum, ed., 'Working Notes: AAAI Fall Symposium on Communicative Action in Humans and Machines', American Association for Artificial Intelligence, Menlo Park, California, pp. 28–35.

Dowty, D. R., Wall, R. E. & Peters, S. (1981), *Introduction to Montague Semantics*, Springer.

Dutoit, T. & Styliannu, Y. (2003), Text-to-Speech Synthesis, in *The Oxford Handbook of Computational Linguistics* (Mitkov 2003), pp. 323–3338.

Elhadad, M. (1993), FUF: The Universal Unifier – User Manual. Version 5.2., Technical report, Ben Gurion University of the Negev.
URL: *http://www.cs.bgu.ac.il/surge/index.html*

Ferguson, G., Allen, J. & Miller, B. (1996), TRAINS-95: Towards a Mixed-Initiative Planning Assistant, *in* 'AIPS-96'.

Flucht-Eriksson, A. (1999), 'A survey of knowledge sources in dialogue systems', *Electronic Transactions on Artificial Intelligence* **3**, 5–32.

Frank, A., Kriegera, H.-U., Xua, F., Uszkoreita, H., Crysmanna, B., Jorga, B. & Schäfer, U. (2007), 'Question answering from structured knowledge sources', *Journal Of Applied Logic* **5**, 20–48.

Gaasterland, T., Parke, G. & Minker, J. (1992), 'An overview of cooperative answering', *Journal of Intelligent Information Systems* pp. 123–157.

Gazdar, G. & Mellish, C. (1989), *Natural language processing in Prolog: An introduction to computational linguistics*, Addison-Wesley.

Glass, M. (2001), 'Processing language input in the CIRCSIM-Tutor intelligent tutoring system', *aied* .

Glass, M. S. (1999), Broadening input understanding in a language-based intelligent tutoring system, PhD thesis, Illinois Institute of Technology.

Graesser, A. C., Person, N. K. & Harter, D. (2001), Teaching tactics and dialogue in AutoTutor, *in* 'Artificial Intelligence in Education'.

Graesser, A. C., VanLehn, K., Rose, C. P., Jordan, P. W. & Harter, D. (2001), 'Intelligent tutoring systems with conversational dialogue', *AI Magazine* .

Grice, H. P. (1975), Logic and Conversation, *in* 'Syntax and Semantics III: Speech Acts', Academic Press, pp. 41–58. Reprinted in (Kasher 1998), vol. 5, pp. 145–161.

Grosz, B. J., Sparck Jones, K. & Webber, B. L. (1986), *Readings in Natural Language Processing*, Morgan Kaufmann.

Haarslev, V. & Moller, R. (2001), RACER System Description, *in* 'IJCAR '01: Proceedings of the First International Joint Conference on Automated Reasoning'.

Haller, S. (1999), 'An introduction to interactive discourse processing from the perspective of plan recognition and text planning', *Artificial Intelligence Review* **13**, 259–311.

Harabagiu, S. & Moldovan, D. (2003), Question Answering, in *The Oxford Handbook of Computational Linguistics* (Mitkov 2003), pp. 560–582.

Hirschman, L. & Gaizauskas, R. (2001), 'Natural Language Question Answering: The View From Here', *Natural Language Engineering* **7**(4), 275–300.

Hobbs, J. R., Appelt, D., Bear, J., Isreal, D., Kaneyama, M., Stickel, M. & Tyson, M. (1996), *FASTUS: A cascaded finite-state transducer for extracting information from natural-language text*, MIT Press, pp. 383–406.

Jackson, P. & Moulinier, I. (2002), *Natural language processing for online applications: text retrieval, extraction and categorization*, John Benjamins.

Johnson, W. L., Rickel, J. W. & Lester, J. C. (2000), 'Animated Pedagogical Agents: Face-to-Face Interaction in Interactive Learning Environments'.

Jordan, P., Ringenberg, M. & Hall, B. (9999), 'Rapidly developing dialogue systems that support learning studies'.

Jordan, P. W., Hall, B., Ringenberg, M., Cue, Y. & Rose, C. (2007), Tools for authoring a dialogue agent that participates in learning studies, *in* 'Proceedings of Artificial Intelligence in Education'.

Jurafsky, D. & Martin, J. H. (2000), *Speech and language processing: an introduction to natural language processing, computational linguistics and speech recognition*, Prentice Hall.

Kacmarcik, G. (2005), Question-answering in role-playing games, *in* 'AAAI Workshop on Question Answering in Restricted Domains'.

Kacmarcik, G. (2006), Using natural language to manage NPC dialog, *in* 'Proceedings of the Second Artificial Intelligence and Interactive Digital Entertainment Conference'.

Kamp, H. & Reyle, U. (1993), *From discourse to logic: Introduction to the modeltheoretic semantics of natural language, formal logic, and discourse representation theory*, Springer.

Kasher, A. (1998), *Pragmatics: Critical Concepts*, Routledge.

Kay, M. (1980), Functional Unification Grammar: A Formalism for Machine Translation, *in* 'COLING'.

Koller, A., Debusmann, R., Gabsdil, M. & Striegnitz, K. (2004), 'Put my Galakmid coin into the dispenser and kick it: Computational linguistics and theorem proving in a computer game', *Journal of Logic, Language and Information* **13**(2), 187–206.

Kopp, S., Krenn, B., Marsella, S., Marshall, A. N., Pelachaud, C., Pirker, H., Thorisson, K. R. & Vilhjálmsson, H. (2006), Towards a common framework for multimodal generation, *in* 'Intelligent Virtual Agents', Springer.

Kwok, C. C. T., Etzioni, O. & Weld, D. S. (2001), Scaling Question-answering to the web, *in* 'World Wide Web'.

Landaur, T., Foltz, P. W. & Laham, D. (1998), 'Introduction to Latent Semantic Analysis', *Discourse Processes* **25**, 259–284.

Lane, H., Core, M. G., Gomboc, D., Karnavat, A. & Rosenberg, M. (2007), Intelligent tutoring for interpersonal and intercultural skills, *in* 'Proceedings of I/ITSEC'.

Larsson, S. & Traum, D. (2000), 'Information state and dialogue management in the TRINDI dialog move engine toolkit', *Natural Language Engineering* **6**, 323–340.

Lee, B. (1987), *Case studies in business computing*, Computer Studies Series, Hutchinson.

Leech, G. & Weisser, M. (2003), Pragmatics and Dialogue, in *The Oxford Handbook of Computational Linguistics* (Mitkov 2003), pp. 136–156.

Levin, E., Narayanan, S., Biatov, K., Bocchieri, E., DiFabbrizio, E., Lee, S., Pokrovsky, A., Ruscitti, P., Rahim, M. & Walker, M. (2000), The AT&T-DARPA communicator mixed-initiative spoken dialogue system, *in* 'ICSLP2000, 6th International Conference on Spoken Language Processing', Beijing, China.

Litman, D. J. & Silliman, S. (2004), *in* 'Proceedings of the Human Language Technology Conference: 4th Meeting of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL), Boston, MA.'.

Marsella, S. C., Johnson, W. L. & LaBore, C. (2000), Interactive pedagogical drama, *in* 'Proceedings of the Fourth International Conference on Autonomous Agents'.

McRoy, S. W., Haller, S. W. & Ali, S. S. (1997), 'Uniform Knowledge Representation for NLP in the B2 System', *Natural Language Engineering* **3**(2), 123–145.

Mitkov, R. (2003), *The Oxford Handbook of Computational Linguistics*, Oxford University Press.

Molla, D. & Vicedo, J. L. (2007), 'Question Answering in Restricted Domains: An Overview', *Computational Linguistics* **33**(1), 41–62.

Olney, A., Louwerse, M., Matthews, E., Marineau, J., Hite-Mitchell, H. & Graesser, A. (2003), Utterance classification in AutoTutor, *in* 'Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing'.

Pelachaud, C. & Bilvi, M. (2003), Computational Model of Believable Conversational Agents, Springer.

Pereira, F. C. N. & Schieber, S. M. (1987), *Prolog and Natural Language Analysis*, Vol. 10 of *CSLI Lecture Notes Series*, Center for the Study of Language and Information.

Popescu, A.-M., Etzioni, O. & Kautz, H. (2003), Towards a theory of natural language interfaces to databases, *in* 'Proceedings of the 8th international conference on Intelligent user interfaces'.

Popescu, O., Aleven, V. & Koedinger, K. (2004), Understanding students' explanations in geometry tutoring, *in* 'Proceedings of the 20th international conference on Computational Linguistics'.

Schäfer, U. (2006), Integrating Deep and Shallow Natural Language Processing Components - Representations and Hybrid Architectures, PhD thesis, Universitat des Saarlandes.

Schwitter, R., Rinaldi, F., Dowdall, J. & Hess, M. (2004), NLP for answer extraction in technical domains, *in* 'EACL workshop on Natural Language Processing for Question Answering'.
URL: *http://staff.science.uva.nl/ mdr/NLP4QA/*

Searle, J. R. (1975), Indirect Speech Acts, *in* 'Syntax and Semantics III: Speech Acts', Academic Press, pp. 59–82. Reprinted in (Kasher 1998), vol. 2, pp. 48–64.

Traum, D. R. & Allen, J. F. (1994), Discourse obligations in dialogue processing, *in* 'Proceedings of the Thirty-Second Meeting of the Association for Computational Linguistics'.

Voorhees, E. N. & Dang, H. T. (2005), Overview of the TREC 2005 Question-Answering Track, *in* 'TREC QA track'.

Wallace, R. (2001), 'Elements of AIML style'.

Ward, R. (1998), 'Active, collaborative and case-based learning with computer-based case scenarios', *Computers in Education* **30**(1/2), 103–110.

Warren, D. H. & Pereira, F. C. N. (1982), 'An Efficient Easily Adaptable System for Interpreting Natural Language Queries', *American Journal of Computational Linguistics* **8**(3/4).

Weizenbaum, J. (1966), 'ELIZA – A Computer Program For the Study of Natural Language Communication Between Man and Machine', *Communications of the ACM* **9**, 36–45.
URL: *http://i5.nyu.edu/ mm64/x52.9265/january1966.html*

Wilensky, R., Chin, D. N., Luria, M., Martin, J., Mayfield, J. & Wu, D. (2001), 'The Berkeley UNIX Consultant Project', *Artificial Intelligence Review* **14**(4), 43–88.

Winograd, T. (1972), *Understanding Natural Language*, Academic Press.

Woods, W. A. (1973), Progress in natural language understanding: an application to lunar geology, *in* 'AFIPS Conference Proceedings'.

Yeates, D. & Wakefield, T. (2004), *Systems Analysis and Design*, 2 edn, Prentice Hall.