ANNOTATION OF JOINT PROJECTS AND INFORMATION STATES IN HUMAN-NPC DIALOGUES

Núria Bertomeu
Anton Benz
Zentrum für Allgemeine Sprachwissenschaft

Abstract
We present a corpus of human-NPC interactions in a virtual environment. The corpus has been obtained through a Wizard-of-Oz experiment simulating a scenario where the user furnishes a room with the help of a virtual interior designer. With the aim of extracting useful information for the development of a dialogue model, an annotation scheme and representation format have been designed. The unit of annotation is the minimal joint project. Minimal joint projects are represented as feature-structures containing information on their goals, the information state shared by the dialogue participants and the actions composing the projects. The representation format is suitable for describing dialogues independently of the task and domain and can serve as representation of dialogue states in dialogue models. A methodology for the generation of project representations relying on manual and automatic annotation has also been developed. The annotated corpus allows for the automatic extraction of dialogue transitions and delivers useful information for the development of the natural language understanding and generation modules.

Keywords: dialogue, NPC, corpus, information state, joint project

I. INTRODUCTION

In the latter decade, Massive Multiplayer Online Role Playing Games and virtual reality environments, such as Warcraft and Second Life, are proliferating. In such games players cohabit a 3D-environment with other players. To populate such 3D-worlds in the early stages of the game's existence and to fill roles which humans do not want to play, such as waiter, shop assistant, etc. non-player-characters (NPC) are created. NPCs are virtual characters, who can serve different purposes in the game, ranging from providing information to helping the user to carry out some task. However, the linguistic capabilities of NPCs are currently very limited. There are very few commercial games which handle linguistic input[1].

Recently, there has been some research on providing virtual characters in 3D-worlds with conversational capabilities. The NICE fairy-tale game (Gustafson et al., 2005) and the Mission Rehearsal Exercise System (Hill et al., 2003) are some of the most sophisticated resulting prototypes. Still a lot of research needs to be done to provide NPCs with natural language dialogue capabilities which enhance the naturalness of the dialogue. These capabilities may include e.g. the understanding of implicatures and the ability of producing pragmatically adequate responses. Such research requires first of all data to analyze how humans interact with NPCs. Although there exist some corpora of interactions between humans, mostly children, and virtual agents, e.g. Narayanan and Potamianos (2002), Gustafson et al. (2005), data about human-NPC interactions in 3D-environments are still rather scarce.

In this paper we present a corpus of human-NPC interactions, obtained by means of a Wizard-of-Oz (WoZ) experiment. The interactions take place in a virtual reality environment, where both an adult subject and the simulated NPC are present in the same room through their

avatars. The NPC's role is the one of an interior designer/furniture saleswoman who helps the subject furnishing a living-room. Her main tasks are to find out about the preferences of the subject, present adequate objects according to these preferences and place them in the room according to the instructions of the subject.

Our investigation of the data aims at addressing questions relevant for the development of dialogue models for NPCs, e.g. which action should an NPC carry out given a particular context. For this, we need to annotate not only the actions performed by the dialogue participants (DPs), but also the changes that these actions produce in the information state shared by them. We will use here the term *information state* to denote the information which has been established during the dialogue: concretely, the parameter values already fixed and the parameter values under discussion and under consideration, similar e.g. to Ginzburg's Dialogue Gameboard (Ginzburg, 1995). It should not be confused with the larger notion of Information State (IS) in the Information State Update (ISU) framework, e.g. (Larsson & Traum, 2000). An IS represents the information that the DPs have at a particular point in the dialogue, including not only the content of the dialog conveyed so far, but also the latest contribution to the dialogue, the immediate actions to carry out by the DPs, as well as the long term plans motivating these actions. Our information state, since it only represents the content of the dialogue conveyed so far, corresponds only to a part of an IS representation of dialogue context.

There exist several coding schemes for the annotation of dialogue acts in task-oriented dialogues. Some of the most popular are the Dialog Act Mark-up in Several Layers (DAMSL) (Core & Allen, 1997) and the HCRC (Carletta & Isard, 1996) schemes. DAMSL is a general multi-dimensional scheme, a sub-set of whose dimensions and functions we have adopted in our scheme. The HCRC interestingly incorporates the annotation of adjacency pairs (called *games* in the scheme), which will also be present in our annotation.

Regarding the annotation of information states, Poesio et al. (1999) have carried a pilot study for the annotation of ISs, concluding that these are not suitable for large-scale annotation, because the task is time-consuming and difficult. Georgila et al. (2005) have automatically annotated ISUs in the COMMUNICATOR corpus. However, since the content of information states is domain and task-specific such a procedure is not easily transferable to our corpus.

In this paper we present an annotation scheme and representation format for dialogue states. The main considerations when designing them were that they should be general enough to describe dialogues in other domains and tasks. A second consideration was that they should allow for capturing larger phases in the interactions, but be kept flexible to handle interruptions and the development of topics in parallel. This latter consideration brought us to take a bottom-up approach to the analysis by choosing as our annotation unit *minimal joint projects* (Clark, 1996). Minimal joint projects are adjacency pairs which have a purpose and carry out an update of the information state. They will be represented in feature-structures which contain information on their function, goal, information state and the actions building them up. The actions are further specified according to the act they perform and their role in the project, among other information. The annotator does not have to annotate the information states directly, these will be extracted from other information easier to annotate and will be presented to the annotator for correction. These project representations allow us to extract useful information for the dialogue model. They can also be used in the dialogue model as state representations, based on which network transitions can be formulated in finite-state dialogue models.

Finally, the paper is structured as follows: the next section describes the WoZ-experiment with which we collected the corpus, and gives an overview of the main characteristics of the data. Section 3 explains the theoretical background behind the

annotation scheme, presents the scheme and briefly describes the process of annotation. Section 4 discusses some of the research questions that can be addressed with the annotated corpus. Finally, section 5 summarizes and concludes.

II. THE CORPUS

In order to gather data on how humans interact with NPCs in the graphical environments of computer-games and virtual realities, we carried out a WoZ-experiment. In a WoZ-experiment a person plays the role of the system to be simulated and subjects interact with a system interface, unaware that there is a human behind it. In the following subsections we describe our experiment and the data obtained.

II.1. Experimental setting

The experiment took place in the virtual environment Twinity[2], which provides a virtual representation of the city of Berlin. In this virtual world people can own apartments and furnish them as they wish, by purchasing virtual furniture and decoration objects. The furniture pieces can be chosen from a catalogue and moved around in the apartment, so that different locations for them can be tried.

Given this possibility of owning and furnishing a virtual apartment, it seemed to us practicable and desirable to model an NPC with the role of an interior designer/furniture saleswoman who helps people furnishing their apartments by showing them objects which match their wishes, trying different locations for these objects and giving advice.

The task of the subjects was, thus, to furnish a living-room with the help of the interior designer/furniture saleswoman. The subject and the Wizard were sitting spatially separated in different rooms although they both were present as avatars in the virtual room and could see the changes made to it in real time. The interaction took place in typed English language through the chat-interface provided by Twinity. This chat-interface presents the contributions of each dialogue participant in speech balloons. Figure 1 shows a snapshot of one of the sessions.

Figure 1: Interaction between the NPC Alexandra and a subject

In the preparation phase of the experiment the Wizard was instructed about decoration principles, such as colour combinations and styles, and got very well acquainted with the range of available living-room furniture and decoration. She was told to follow a specific behaviour protocol, but still to react flexible to the initiative of the subject. The protocol stated the order in which she should address the different types of objects, as well as the different steps to be taken within the discussion of a specific type of object. For each step she had a collection of readymade utterances which she should use if possible. These ready-made utterances were paired with short codes in a text substitution program, which allowed to automatically type them in the chat window by typing the codes. This resulted in a reduction of the typing time and typos and contributed to the impression of machine-likeness.

An important reason for using a behaviour protocol is that we wanted to obtain comparable data among the subjects. If the Wizard behaves with total freedom there is no way to know whether the different behaviour of the subjects is due to intrinsic variation among them or to the different behaviour of the Wizard.

The subjects were given written instructions and a video of a sample interaction. In order to increase their motivation they were offered an economic incentive: the half of the best rooms would be awarded with a prize.

Eighteen subjects took part in the experiment. The interactions lasted one hour. They were all logged and video-recorded. Afterwards, the subjects were asked to fill a form where we assessed the credibility of the experiment. Sixteen subjects believed to be interacting with an NPC, although four of them sometimes had the impression of being interacting with a person. Fifteen subjects found the dialogue useful and easy to carry out.

II.2. Main characteristics of the data

As a result of the WoZ-experiment we obtained a corpus of mixed-initiative human-NPC interactions consisting of 18 dialogues, 23.015 alpha-numeric strings, 4.313 utterances and 3.171 turns.

As mentioned above, the main goal of the dialogues is to choose and place instances of certain types of objects, such as a sofa, a coffee-table, an arm-chair, etc. Hence, progress in the task fulfilment is achieved by addressing the following parameters:

(1)

- family of objects (furniture, accessories, electro, decoration),
- object type (e.g. sofa, sideboard, plant, TV),
- property-value pairs of the object (e.g. colour: red, material: leather, size: large),
- object (e.g. "Sofa Consuelo", "Sofa Isadora", "Chair Beverly"),
- location of the object (e.g. "in front of the sofa", "between the windows", "on the top of the shelf"),
- quantity of instances of the object for certain types of objects (e.g. "4 chairs", "2 arm-chairs").

The main proceeding of the Wizard was to find out whether the customer wants a certain type of object in the room and if it is so, to choose the corresponding object and its location, and eventually decide on the quantity of instances of the object. A number of sub-goals serve these main purposes. Below we find a list of main goals and sub-goals linearly and hierarchically represented[3].

(2)

- **introduce family of objects**
- **introduce object type (find out whether the user wants it)**
- **choose object**
  - determine consideration set
    - determine value for property
  - examine alternative proposed
  - select alternative
- **choose location**
  - determine location
  - examine location
  - select location
- **choose quantity**
  - determine quantity
  - examine quantity
  - select quantity
- **close discussion about object type**
- **close discussion about family of object types**

However, not all of these sub-goals need always to be carried out. Goals are sometimes carried out just implicitly. A single utterance may address several goals and there is only a partial order in which the goals need to be pursued.

Since the task is to build up an arrangement of furniture and decoration, the choice of an object and its location has to fit with the other objects and their locations. A consequence of this is that the different types of objects are often not dealt with one after the other in an orderly fashion. The discussion of an object type may be interrupted and another object type

may be addressed, several object types may be addressed in parallel and closed object types may be raised again.

The following dialogue gives a glimpse of the data:

(3) USR.1: And do we have a little side table for the TV?
NPC.1: I could offer you another small table or a sideboard.
USR.2: Then I'll take a sideboard that's similar to my shelf.
NPC.2: Let me check if we have something like that.
NPC.3: What about this one?
USR.3: No, that doesn't fit in here.
NPC.4: Do you want me to show you another one?
USR.4: Yes, I'd like to.
NPC.5: Do you like it?
USR.5: Is there a black or white sideboard?
NPC.6: No I'm afraid not, they are all of light or dark wood.
USR.6: Ok, then I'll take this one.
NPC.7: All right.

In this example the user takes the initiative by requesting to address a certain object type, a *side table*. The NPC initiates a side-sequence by requesting information on which object type the user concretely wants to address, whether a *sideboard* or a *small table*. The user replies that she wants a *sideboard* and requests that this be *similar to her shelf*, restricting, thus, the consideration set. The NPC offers an alternative, which is simply rejected by the user. The NPC reacts by proposing to show another alternative, which the user accepts. The alternative is shown, but the NPC does not accept or reject it immediately. She initiates a side-sequence by requesting a sideboard in *black* or *white*, addressing, thus, the consideration set again. The NPC rejects the request, since there are no sideboards with such characteristics and she proposes *light* or *dark wooden* sideboards. This information helps the user to make a decision, which she communicates by accepting the last alternative presented. Finally, the NPC acknowledges the decision.

This example shows that, though the limited nature of the simulated scenario and the concrete behaviour pattern followed by the Wizard, the interactions exhibit quite a high degree of freedom and contain many complex phenomena.

III. CORPUS ANNOTATION AND REPRESENTATION

In this section we present an annotation scheme and representation format. Since our main goal when investigating the data is to address questions relevant for the development of a dialogue model, such as which decisions are made in which contexts, we need to annotate both the actions carried out in the interactions and the changes that these actions make to the shared information state of the DPs. A second consideration is that the scheme and representation should be portable to other tasks and domains, which can be achieved by describing the data with general attributes, which may take task or domain-specific values. Finally, the scheme and representation should be able to capture how higher goals are pursued, but still be flexible enough to cover the parallel handling of issues, interruptions, etc.

III.1. Theoretical background

Keeping in mind these considerations we have taken a bottom-up approach to analyzing the data by annotating the minimal units that carry out an update of the information state of the

DPs. This minimal unit is the *project*, as in Clark (1996). A minimal joint project has a purpose and is usually realized by an adjacency pair. An adjacency pair consists of two ordered actions carried out by different agents. The first action initiates the joint project, by raising an issue, and the second action completes it.

According to Clark (1996), minimal projects can be related with each other in different ways. One of them is *chaining*. In chaining the completing act of the project initiates itself another project which is then completed by a third act. This is the case for question-answer pairs, where the answer raises a new issue, which needs to be accepted. Here follows an example:

(4) NPC: Which colour do you want for the walls?
USR: I want them slightly off white.
NPC: Ok.

Joint projects can also be embedded in other projects. Embedded projects can be concerned with dialogue-management, as in the following example:

(5) NPC.1: Would you like to see a footstool?
USR.1: *What is a footstool?*
NPC.2: *A footstool is a support for the feet.*
USR.2: No, thank you. I want to see an arm-chair.

They can also build up side-sequences exchanging some information necessary for completing the embedding project. Here follows another example:

(6) NPC.1: Which colour would you like the walls to be?
USR.1: *Are they white now?*
NPC.2: *Yes.*
USR.2: Then I'll leave them like this.

Projects can also be preceded by other projects which prepare them. These introductory projects are called pre-sequences or preliminary projects. The main project in (7) is headed by a preliminary, which addresses the addressing of a particular issue.

(7) NPC.1: *Let us start with considering what you would like to do in this room.*
USR.1: *Ok.*
NPC.2: Is it going to double as a dining-room or just used to relax and entertain your friends?

Joint projects have a function in dialogue. They can be concerned with bringing the task forward, with managing the dialogue, with meta-talk, etc. As we already mentioned, projects serve some purpose, so both the initiating and completing acts contribute to the achievement of this purpose. In (4), for example, the purpose is to determine the value for the property *colour* of the walls.

Task-related projects fix certain pieces of information, or parameters. This information fixed during the dialogue is shared by the DPs. In our dialogues fixed information can have different qualities: it can be part of what is established in the dialogue to hold of the room (a set of beliefs about the room as the dialogue proceeds), it can belong to the set of alternative objects under consideration (alternatives accepted, but about which no final decision has been made), and it can be the object type and family of object types currently under discussion. These three components build up an information state. According to this, task-related joint projects are initiated in a given information state and after their completion the information state is updated, so we can speak of a unique information state for the joint project. For example, for the project in (4) the information state is such that it is not known which colour the customer wants the walls to be. After the project, it is fixed that the colour should be white.

III.2. Project description and representation

We describe a task-related project by means of its function, goal and information state. If the project contains an embedded project or is introduced by a pre-sequence we also include the embedded project or the pre-sequence in the description of the project. Instances of chaining will not be annotated as embedded projects, but as projects consisting of three actions. The individual actions which constitute the project are further described by means of the speaker contributing them, the act performed, their role in the project (initiating vs. completing), their status as linguistic or physical action, their status as implicit or explicit action, the time when they were contributed and the set of parameters they raise and discuss.

Since the information we want to code is sometimes quite complex, we have decided to code the project as a complex feature-structure. Feature-structures are objects described by means of attribute-value pairs, for which the values can be themselves complex feature-structures. A feature-structure representation allows for a clear visualization of how the different parts of the project hang together. For the coding of feature-structures in XML format we have followed the guidelines of the Text Encoding Initiative (TEI)[4].

Figure 2 shows the feature-structure representation of the project in (8):

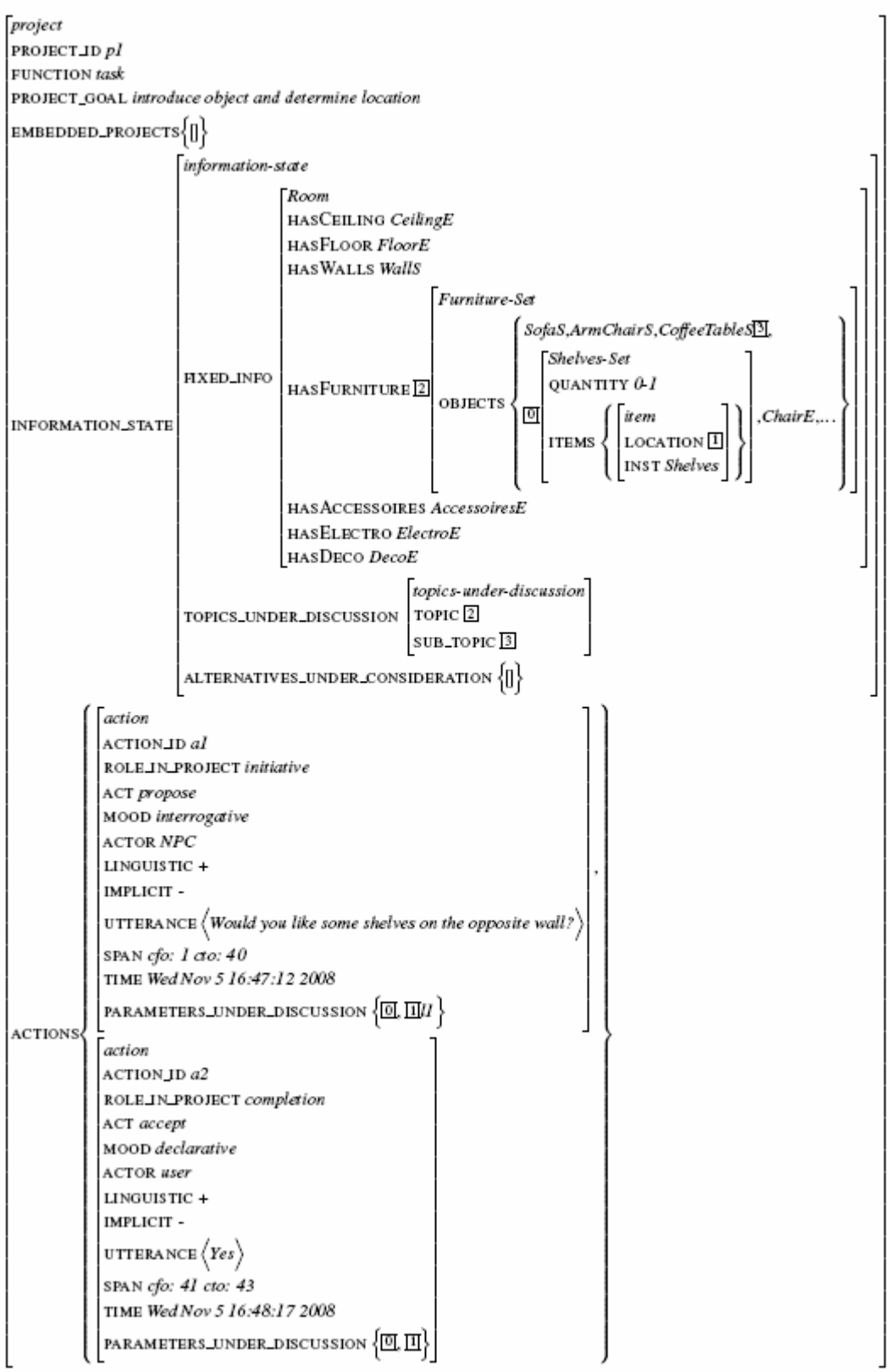

Figure 2: Project representation

(8) NPC: Would you like some shelves on the opposite wall?
USR: Yes.

Let us now look in detail at the features which require an explanation:

- FUNCTION: The project can carry out one of the following functions: *task, dialogue management, preliminary, side-sequence, greet, farewell* and *other*.
  As we explained, task-related projects update the information state. Projects with any of the other functions do not. The project in (8) is an example of a project with a task-related function. Projects with a dialogue-management function are concerned with the grounding of a previous utterance. They ask or provide clarification for the intended meaning of the previous utterance, ask about its understanding or acknowledge this. An example of a project concerned with dialogue management is provided in (5). Projects with a side-sequence function carry out an exchange of information relevant for completing the project embedding them, as illustrated by the example in (6). Preliminary projects are concerned with announcing or requesting the initiation of a task-related project, as in (7). Projects concerned with meta-talk are about the processing carried out by the system. Typical examples are requests by the system for the user to wait. Projects may also have a greeting or farewell function if they are concerned with greeting and the presentation of the DPs, or with saying goodbye, respectively. Finally, projects not carrying out any of the above mentioned functions, e.g. comments, will be annotated as pursuing a function *other*.
- PROJECT GOAL: Only task-related projects will be specified for a project goal. From a first observation of the data we have identified the abstract goals in (2). The goals are composed by an action and a type of parameter. Each project carries out one or several minimal goals, that is, those goals at the lowest level of nesting in (2). In (8), for example, the goals are *introduce object type* and *determine location*.
- EMBEDDED PROJECTS: Projects embedding other project(s) or being introduced by preliminaries will have as value for this feature a set containing the embedded or preliminary project(s). Embedded and preliminary projects are represented as feature-structure objects similar to normal projects but with a reduced set of features. For example, they are not specified for an information state, since they are not supposed to update it and the information state of the embedding project holds for the embedded or preliminary project as well.
- INFORMATION STATE: As already mentioned, the information state will consist of three blocks of information, which differ in their status between: fixed, under consideration and under discussion. The information state represents the values for the parameters presented in (1) which are valid when the project takes place. The completing act changes this information state. However, the change is first represented in the information state of the subsequent project. This leads to a less verbose annotation. If we annotated an information state after the initiating act and then after the completing act, we would have to annotate the same information state twice. Annotating only the information state previous to the completion of a project still allows us to recover the updated information state, since it is the information state of the subsequent project.

  Let us look now at the different components of the information state:

  - FIXED-INFORMATION: Fixed parameters are represented within a feature-structure representation of the room being arranged. This room representation is analogous to the room specification in the ontology underlying a potential dialogue system. It functions as a template with slots for all the possible furniture and decoration pieces which can be placed in the room and is being filled as changes in the room are being made. It serves both as memory for what has been achieved and as a guide for what still needs to be done. Expressed preferences in terms of property-value pairs about objects for which no instance has been found yet, will also be placed in the room representation

and will provide, thus, an underspecified representation of the object, which will be fully specified once an instance is found.

Following the TEI guidelines, to avoid verbosity in the annotation, objects which are not currently under discussion, whether instantiated or empty, will be represented with a reference. Their full representation will be stored in a library. The reference can be unfolded and, hence, the full representation can be visible in the annotation when necessary.

- TOPICS-UNDER-DISCUSSION: This feature corresponds to a stack of open topics currently under discussion, where the *topic* corresponds to the family of object types and the *sub-topic* to the object type. The top of the stack is not represented here, but in the individual actions, since each action may update it.
- ALTERNATIVES-UNDER-CONSIDERATION: This feature has as value a set of parameters of the same type for cases in which alternatives are accepted by the customer, but he nevertheless requests to see further alternatives and postpones a decision.

To illustrate the information state let us look at Figure 2. Before the completion of the project in (8), the feature FIXED-INFORMATION has as value a template of the room being arranged in which no cover for the ceiling or the floor has been chosen[5]. No objects of the families accessories, electro and decoration have been chosen either, but a cover for the walls has. The topic of the dialogue is currently the set of furniture pieces, which we can see by the coindexation of the TOPIC feature with the *Furniture-Set* in the room representation. Within the Furniture-Set we see that a sofa, an armchair and a coffee-table have been chosen, but a chair and other pieces of other types have not. The coffee-table is still the sub-topic under discussion, as shown by the coindexation, since the new sub-topic proposed in the initiating act has not yet been accepted for discussion. The feature-structure representation of the new sub-topic, however, is already unfolded. As we will see shortly, the parameters under discussion of the initiating and completing acts correspond to certain aspects of this new object.

An object-type is represented by a feature-structure of type *Set*, in this case, the *Shelves-Set*. This feature-structure is specified for a quantity, in this case the quantity of shelves in the room, and a set of items, in this case, all the individual shelves in the room. An item is further described as having a location and an instance, that is, a concrete piece of furniture. Note that at this point the quantity of instances is underspecified between 0 and 10, which means that it is still not known whether the customer wants some shelves. The value of the feature INST is the general abstract type *Shelves*, of which all the shelf models are sub-types and the individual shelves instances.

Figure 3 shows the updated information state after the completion of the project. This information state is part of the following project. As you can see, the shelves have become the sub-topic of the interaction, the quantity of shelves has been fixed to 1 and the location to the location proposed in the preceding project, *l1 (on the opposite wall)*.

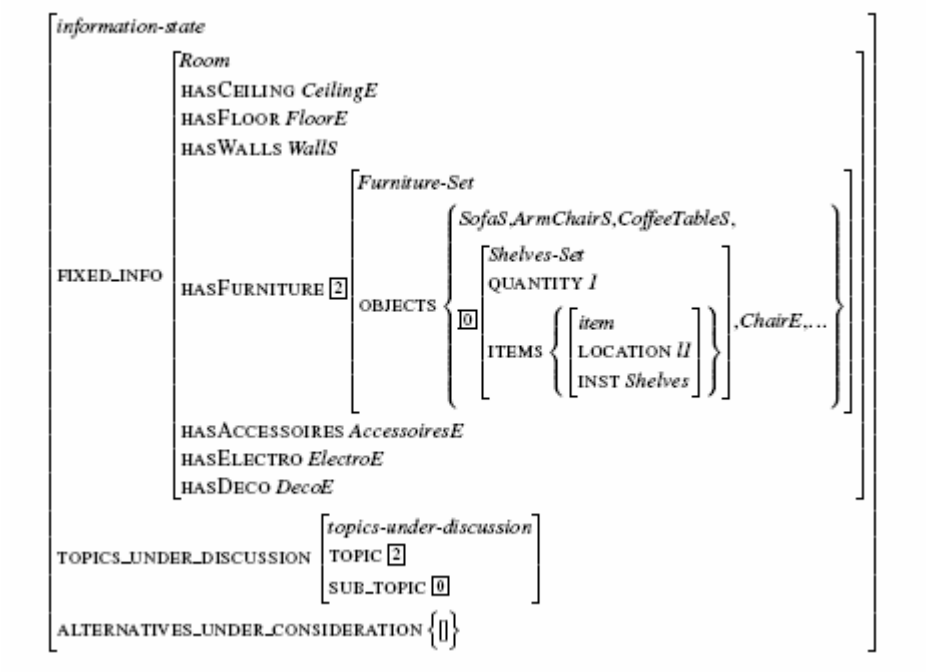

Figure 3: Updated information state

- ACTIONS: This feature has as value the set of actions building up the project. Actions are the minimal units contributing to the achievement of a purpose. They are further specified as follows:
    - ROLE IN PROJECT: An action can be the *initiative* if it is the first action of the project, which raises an issue. The second action of the project is the *completion*.
    - (SPEECH) ACT: Only the actions in projects with a task, dialogue-management, preliminary, side-sequence and meta-talk function will be specified for the attribute ACT. Considering the kind of reaction that they require from the other dialogue participant in our particular task, i.e. the perlocutionary act, we have come up with a reduced set of acts, which nevertheless are sufficient to classify all actions that we have encountered:
      a. **propose:** Proposals are mostly performed by the NPC, since she proposes topics, values for properties, objects, which are subject to acceptance or rejection by the customer. Here follows an example:
      (9) NPC: Maybe you would like to see a black leather arm-chair.
      b. **request:** Requests are mostly performed by the customer, since he wants the designer to do something for him, such as addressing a certain topic at a certain time, showing him objects with certain features, placing objects at certain places, etc. Here follows an example:
      (10) USR: I'd like a black shelf if you have one.
      c. **request info**: Requests for information are mostly, but not only, performed by the NPC in order to find out about the features of the objects that the customer would like to add to the consideration set, or to find out about the location or quantity of the objects. These are usually alternative questions or wh-questions. Here follows an example:
      (11) NPC: Where do you want me to put it?
      d. **accept**: In order to being pursued, both requests and proposals first need to be explicitly or implicitly accepted by the other dialogue participant.

While the customer is free to accept or reject an NPC's proposal, the NPC always will accept the customer's requests if these can be fulfilled. Answers to open questions or answers to alternative questions which deviate from the alternatives proposed also have to be accepted. Here follows an example:

(12) NPC: Would you like to make your sofa the focal point of the room by choosing some bright colour like red?
propose
USR: Yes.
accept

e. **reject:** Requests and proposals can also be rejected. Sometimes this happens only implicitly. Again, requests by the customer are not likely to be rejected if it is not the case that they cannot be fulfilled. Here follow some examples:

(13) NPC: Do you like this one?
propose
USR: No.
reject

f. **provide info:** Mostly only the customer provides information about his preferences upon request by the NPC. Here follows an example:

(14) NPC: Would you like a plain lamp or an eye-catcher?
request info
USR: An eye-catcher.
provide info

g. **acknowledgement:** Actions carrying out acknowledgement build up projects only concerned with dialogue management. They communicate that the dialogue participant has understood and grounded the content of some previous statement. They usually appear in the same form as acceptances ("Ok", "I see", ...), but in different contexts. Here follows an example:

(15) NPC: Will you take it?
propose
USR: Yes. / No.
accept, reject
NPC: Ok.
acknowledgement.

- MOOD: The value for this feature is the sentence mood of the utterance performing the action. Possible sentence moods are: declarative, interrogative and imperative. As you can see in the following examples, illocutionary act and sentence mood are independent from each other and the same illocutionary act can be performed with utterances in different moods:

(16) USR: Do you have a green sofa?
act: request, mood: interrogative
USR: I would like a green sofa.
act: request, mood: declarative
USR: Show me a green sofa.
act: request, mood: imperative

- LINGUISTIC: This is a binary attribute which can have as values "+" or

"-", depending on whether the action has been carried out by an utterance, that is, it performs an illocutionary act, or whether it has been carried out physically, e.g. by showing or placing an object. Here follows as an example a project where the initiating act is performed physically and the completing act is an illocutionary act. Of course, physical actions will not be specified for sentence mood, span and utterance.

(17) the NPC places a sofa
act: propose, linguistic: -
USR: Wonderful!
act: accept, linguistic: +

- IMPLICIT: This is a binary attribute which can have as values + or -, depending on whether the action has been carried out implicitly or explicitly. Implicit actions are carried out by utterances which also carry out another action. The first action will be annotated as implicit. Implicit actions are usually acceptances and rejections. Here follows an example:

(18) NPC: Do you like it?
USR: Don't you have something smaller?
act: reject, implicit: +
act: request, implicit: -

- PARAMETERS UNDER DISCUSSION: This feature has as value the set of parameters under discussion: the issues that the action concretely addresses, i.e. the objects with which the project goals are concerned. From the attentional point of view, it corresponds to the top of the stack of topics under discussion. The different types of parameters (e.g. type of object, location, ...) have been presented in (1). The parameters are coindexed with certain objects in the room representation. With this coindexation it is clear which place they occupy in the room or to which objects they relate. However, this coindexation does not mean that the parameter values under discussion are fixed in the representation of the room. As long as they are under discussion they are not fixed. Only information in the representation of the room which is not under discussion is fixed.

  As an example, look at Figure 2. Here, there are two parameters under discussion: the object type and the location of the object. As value for the parameter object type *Shelves* is proposed, and as value for location on the *opposite wall* is proposed.

  Projects initiated by proposals and requests will have the same parameters under discussion in the initiating and completing acts. However, projects concerned with the exchange of information will have different sets of parameters under discussion in the initiating and completing acts, since the completing act provides a value for the abstract parameter raised in the initiating act.

III.3. The annotation

In order to simplify the complex annotation task at hand, we have come up with a methodology for the annotation of projects in different phases. In the first phase thematic blocks about a single object type have been annotated by creating a single markable containing all the utterances around the object type and choosing the topic and the sub-topic

from the different families of object types and object types presented in (1). This annotation has been carried out with MMAX2[6] (Müller & Strube, 2006). The second part of the annotation is also being carried out with MMAX2. Previous to it, markables corresponding to actions have been automatically created according to punctuation. The annotator, however, can choose to keep these markables as they are, to modify them or to create new ones. The sentence mood has also been automatically annotated according to punctuation, giving the annotator the possibility of correcting it latter. The task of the annotator is then to group the actions in projects and annotate the nesting relations. The function and goal of the project as well as the different illocutionary acts carried out by the actions are also being annotated in this phase. Concrete values for the different parameters under discussion, except for object type and family of object type, which have been annotated in the first phase, are being annotated as well. With all this information almost complete feature-structures are being automatically constructed. Information about the speaker, time stamp and span are also automatically added. In a third phase, the automatically constructed information states are being revised. A reason for this is that sometimes fixed preferences in terms of property-value pairs are implicitly overridden. For all the phases of the annotation carefully written guidelines have been handed to the annotator.

IV. ANALYSIS OF THE DATA

In this section, we briefly discuss several questions which we are planning to address and about which the annotation can give us valuable insights. We are mostly concerned with extracting information useful for developing a finite-state based dialogue model for an NPC playing the role of an interior designer. We are attempting to extract transitions from one dialogue state to another at the most abstract level. Concretely, we want to see what is the context, both at the task and dialogue structure level, in which the NPC chooses to carry out a certain action.

At the task level, for example, we may find out that, when the user has accepted a certain amount of objects for consideration, the NPC should bring him to take a decision by requesting information about which one he likes most. Another transition could be to address the location of the object if this slot is empty and the process of choosing an instance has not started yet or the instance has already been found. Also identifying pairings of information states and input utterances and the sub-sequent information states can help us to formulate information state updates and to restrict the possibilities of interpretation of the input utterance.

From the structural point of view, we want to find out how the pairing of illocutionary acts and sentence moods constrain the generation of responses. For example, a request requires an acceptance or a rejection, however the sentence mood in which the request has been expressed restricts the way in which the acceptance or rejection can be expressed. Here follows an example:

(19) USR: I would like a green sofa.
NPC: Ok. / Here you have. / *Yes. /We don't have any sofa in green. / *No, sorry. / *I'm afraid not.
USR: Do you have a green sofa?
NPC: *Ok. / Here you have. / Yes. / We don't have any sofa in green. / No, sorry. / I'm afraid not.

Further, we want to investigate when projects are chained, that is, the same utterance expresses the completion of the first project (usually implicitly) and the initiative of the following. For example, we find pairs of propose-request, request-propose, request-request

information, etc. It is important that the NPC recognizes the two actions carried out by such utterances, since they both update the information state and request a perlocutionary action from her. It is also important that the NPC is able to generate them, since they contribute to a more natural and less verbose dialogue. Here follows an example:

(20) USR: I would like a green sofa.
request
NPC: What about this one?
accept, propose
USR: I like it.
accept

We also want to identify the different structural realizations of a project with a particular goal, that is, all the possible combinations of initiators and acts, with which the same purpose can be pursued. At present, we have the impression that not all goals can be pursued with the same project internal configurations. For example, while for determining the value of a property all possible combinations of actions and actors are possible, for examining an alternative it seems that the only possibility is that the NPC proposes the object to the customer, who can accept or reject it. This information is important for the dialogue model, since it restricts the possible acts according to the goal to be pursued. It also means that the goals given by the task have an influence on the lowest-level structure of the dialogue, i.e. the internal structure of the project.

Finally, we want to look at how the smallest joint projects that we are annotating build up larger projects around higher-level goals, such as *determine consideration set*, *choose an object*. Our intuition is that it is not always possible to isolate such phases and that phases are sometimes interrupted and sometimes mix with each other in a natural manner. This would support the bottom-up approach that we have taken to analyzing the data, since it provides the necessary flexibility to handle the simultaneous pursuit of several goals, interruptions, etc.

V. SUMMARY AND CONCLUSIONS

We have presented a corpus of human-NPC interactions and its annotation. We have chosen the minimal joint project as our annotation unit, since it is the minimal unit which brings the task a step forward. We have come up with a project representation format in terms of a feature-structure, which allows to easily visualizing the dependencies among the different pieces of information describing the project. The features describing the projects are general, so that the project skeleton can be reused when annotating dialogues with different tasks. We have developed a methodology for coding project representations, based on manual annotation and automatic generation. The obtained project representations are suited for the extraction of dialogue transitions to build up a dialogue model and as dialogue state representations in the dialogue model.

To conclude let us briefly compare our project representations with ISs (Larsson & Traum, 2000). ISs represent the dialogue context of individual moves, while the project represents the context of adjacency pairs. Since much of the information is the same for the whole project, coding projects results in a less verbose corpus annotation. However, the extraction of dialogue states for individual moves from the project representations is straightforward. Project representations only contain information about the preceding dialogue and about what the current actions do. ISs also include information about what is to come in terms of plans and agendas. Coding such information is not straightforward, but by extracting dialogue transitions, concretely, finding out which goals are pursued in which contexts, agendas can be automatically constructed from the project annotations.

Acknowledgements

The research reported in this paper has been conducted as part of the project KomPARSE, carried out in cooperation by the Zentrum für Allgemeine Sprachwissenschaft (ZAS) and the Deutsches Zentrum für Künstliche Intelligenz (DFKI) from June 2008 until June 2011. The project is funded by the ProFIT program of the Investmentbank Berlin and the European Regional Development Fund.

Notes

[1] Lifeline, released by SCEI and Konami, is an example of a game which allows for spoken commands.
[2] See http://www.twinity.com/.
[3] Goals in bold are the main goals, which may include sub-goals.
[4] See http://www.tei-c.org/release/doc/tei-p5-doc/en/html/FS.html.
[5] *E* in the references stands for *empty* object, while *S* stands for *set* object. These labels refer to complete feature-structures stored in libraries.
[6] See http://mmax2.sourceforge.net/.

References

Carletta, J. & Isard, A. (1996). *HCRC dialogue structure coding manual.* Technical report, Human Communication Research Centre, University of Edinburgh.

Clark, H. H. (1996). *Using language.* Cambridge University Press, Cambridge.

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*, pages 28–35, Menlo Park, California. American Association for Artificial Intelligence.

Georgila, K., Lemon, O. & Henderson, J. (2005). Automatic annotation of COMMUNICATOR dialogue data for learning dialogue strategies and user simulations. In *Ninth Workshop on the Semantics and Pragmatics of Dialogue (SEMDIAL: DIALOR)*.

Ginzburg, J. (1995). Resolving questions. *Linguistics and Philosophy*, 18:5, 459–527.

Gustafson, J., Boye, J., Fredriksson, M., Johannesson, L. & Knigsmann, J. (2005). Providing computer game characters with conversational abilities. In *Proceedings of Intelligent Virtual Agent (IVA05)*, Kos, Greece.

Hill, A. W., Gratch, J., Marsella, S., Rickel, J., Swartout, W. & Traum, D. (2003). Virtual humans in the mission rehearsal exercise system. *KI Embodied Conversational Agents*, 17, 32–38.

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

Müller, C. & Strube, M. (2006). Multi-level annotation of linguistic data with MMAX2. In S.

Braun, K. Kohn, and J. Mukherjee (Eds.), *Corpus Technology and Language Pedagogy. New Resources, New Tools, New Methods*., volume 3 of *English Corpus Linguistics*. Peter Lang, Frankfurt.

Narayanan, S. & Potamianos, A. (2002). *Creating conversational interfaces for children. IEEE Transactions on Speech and Audio Processing*, 10, 65–78.

Poesio, M., Cooper, R., Matheson, C. & Traum, D. (1999). *Annotating conversations for Information State Update. In Dialogue.* Amsterdam University.