

GALATEA: A Discourse Modeller Supporting Concept-level Error Handling in Spoken Dialogue Systems

Gabriel Skantze

Department for Speech, Music and Hearing, KTH
Stockholm, Sweden
gabriel@speech.kth.se

Abstract

In this paper, a discourse modeller for conversational spoken dialogue systems, called GALATEA, is presented. Apart from handling the resolution of ellipses and anaphora, it tracks the “grounding status” of concepts that are mentioned during the discourse, i.e. information about who said what when. This grounding information also contains concept confidence scores that are derived from the speech recogniser word confidence scores. The discourse model may then be used for concept-level error handling, i.e. grounding of concepts, fragmentary clarification requests, and detection of erroneous concepts in the model at later stages in the dialogue.

1 Introduction

A common source of errors in spoken dialogue systems is the speech recogniser (ASR), and the handling of such errors is a crucial issue in the design of spoken dialogue systems. The most common way of handling such errors has been to use utterance confidence scores for selecting implicit or explicit verification of full utterances. This often works in dialogues where utterances are relatively short and predictable. However, in dialogue systems that are designed to allow relatively free, conversational speech, with longer, more unpredictable utterances, such utterance-level error handling is often inappropriate. In such systems, statistical n-gram language models are often used in the ASR. These are often better at covering conversational language and tend to degrade more gracefully when the user speaks out of grammar (Knight et al., 2001), but they may also give rise to speech recognition results which are often partly correct. When making semantic interpretations of such results, some semantic concepts will be correct and some not. This calls for error handling on the concept level, i.e. individual concepts in utterances should be

assigned confidence scores and be considered for error handling strategies, such as grounding, clarification and error detection.

In this paper, a discourse modeller for conversational spoken language, called GALATEA, is presented. It is especially designed to support concept-level error handling. GALATEA is not a complete dialogue manager, but rather a processing step in the interpretation process, where utterances are interpreted in context. Apart from handling the resolution of ellipses and anaphora, it tracks the *grounding status* of concepts that are mentioned during the discourse, i.e. information about who said what when. This grounding information also contains concept confidence scores that are derived from the speech recogniser word confidence scores. GALATEA builds a discourse model – a model of what has been said during the discourse, and which entities are referred to. The discourse model may then be consulted by an action manager that selects error handling strategies, such as grounding, clarification and late error detection, on the concept level.

GALATEA is developed within the HIGGINS project (Edlund, et al., 2004). HIGGINS is a test bed for investigating error handling techniques in spoken dialogue systems. The initial domain chosen for HIGGINS is that of pedestrian city navigation and guiding, which is similar to the now classic MapTask (Anderson et al., 1991) domain, as well as to a number of guide systems, such as REAL (Baus et al., 2001). A user gives the system a destination and the system guides the user by giving verbal instructions. The system does not have access to the users’ positions. Instead, it has to rely on their descriptions of their surroundings. Since the user is moving, the system continually has to update its model of the user’s position and provide new, possibly amended instructions until the destination is reached. For simulation, a 3D model of a virtual city is used. The domain is complex enough to generate a variety of error types. The surroundings the user and system talk about contain complex landmarks and relations that are challenging to interpret and represent semantically. For such semantic representations, deep semantic structures are needed.

Table 1 shows a (made-up) scenario with recognition results, which displays the current capabilities of the system¹. The different error handling techniques illustrated in the example will be described later on.

Turn	Transcription	ASR result
S.1	Where do you want to go?	
U.2	To an ATM	TO AN ATM
S.3	Ok, an ATM. Can you describe where you are now?	
U.4	I have a large concrete building on my left and a tree in front of me.	I HAVE LARGE HELLO CONCRETE BUILDING ON MY LEFT AND TREE ON RIGHT
S.5	Which colour does the concrete building have?	
U.6	Red	RED
S.7	Ok, can you see a wooden building in front of you?	
U.8	No, but I have one on my right.	NO I HAVE ONE ON RIGHT
S.9	On your right?	
U.10	Yes	YES
S.11	Ok, I think I know where you are. Walk a little bit forward and take left after the red building.	

Table 1: An example scenario. The rightmost column shows ASR results in which word confidence scores are represented with different shades of grey (dark means high confidence).

2 Error handling in dialogue systems

In this section, issues important for error handling in spoken dialogue systems will be discussed.

2.1 Miscommunication

Miscommunication is a general term that denotes all kinds of problems that may occur in dialogue. A common distinction is made between *misunderstanding* and *non-understanding* (e.g. Hirst et al., 1994). Misunderstanding means that the addressee obtains an interpretation that is not in line with the speaker’s intentions. If the addressee fails to obtain any interpretation at all, obtains more than one interpretation, with no way to choose among them, or does not have enough confi-

¹ The system is currently built for Swedish, but all examples in this paper have been translated into English.

dence in any interpretation, a non-understanding has occurred. One important difference between non-understandings and misunderstandings is that non-understandings are recognized immediately by the addressee, while misunderstandings may not be identified until a later stage in the dialogue. Both of these forms of miscommunication may concern complete utterances or parts of utterances, i.e. partial misunderstanding and partial non-understanding.

One may also classify problems depending on at which “action level” they occur. Clark (1996) makes a distinction between four levels of action that takes place when a speaker is trying to say something to an addressee:

- Acceptance: proposal and consideration.
- Understanding: signalling and recognition.
- Perception: presentation and identification.
- Contact: execution and attention.

For successful communication to take place, communication must succeed on all these levels. The order of the levels is important; in order to succeed on one level, all the other levels below it must be completed.

2.2 Early error detection

Given a speech recognition result, a system must determine if it should accept the utterance and hypothesise that this is what the user (might have) said, or reject it. If the utterance is accepted, some sort of confidence score is often assigned to it. This is the process of *early error detection*. The confidence scores are often based on the probabilities from the acoustic and language models, and the structure of the n-best list (Evermann and Woodland, 2000). To improve early error detection, machine-learning has been used to classify utterances as correct or incorrect, based on features from the recognition result, acoustic features, and dialogue history (e.g. Gabsdil and Lemon, 2004).

For concept-level error handling, confidence scores should be calculated for the individual words in the speech recognition result, just like in Table 1, and transferred into concept confidence scores during semantic interpretation (Gabsdil and Bos, 2003). Skantze & Edlund (2004a) investigates the use of machine learning for early error detection on the word-level, using confidence scores as well as features from the utterance and discourse context.

2.3 Grounding and clarification

Grounding is the process by which speakers establish information as part of common ground well enough for current purposes (Clark, 1996). They do this by giving positive and negative evidence of understanding. If positive evidence is given on one of the action levels

discussed above, all the other levels below it are presumed to have succeeded. If negative evidence is signalled on one of the levels, all the other levels above it are also presumed to have failed, while the ones below it are presumed to have succeeded.

According to Clark, every contribution requires positive evidence, if it is to be regarded as common ground. Clark (1996) discusses different kinds of positive evidence:

- Assertion of understanding.
- Presupposition of understanding.
- Display of understanding.
- Exemplification of understanding.

Assertion of understanding, such as “mhm”, “okey”, “I understand”, is a common evidence. However, it can only give evidence on the utterance level, and it cannot be used to verify that the understanding was correct. Presupposition of understanding means that the addressee continues with a relevant next contribution. The distinction between “display” and “exemplification” is not so clear, but both includes cases where the addressee displays or exemplifies what she has constructed the speaker to mean in the next turn, including verbatim repetitions or paraphrases. I will use the term “display” here for the cases where (parts of) what is said is repeated. By displaying understanding (in this sense), speakers may verify that their understanding is correct, which may also be done on the concept level. Take for example the turn S.5 in Table 1. Apart from requesting the colour of the building, the system also gives some evidence of understanding. The understanding of the concepts CONCRETE and BUILDING are displayed, but not the concept LARGE. If the understanding was incorrect (i.e. a misunderstanding), the user now has the opportunity to correct the system.

If the addressee does not understand, or is not sufficiently confident in parts of her understanding (i.e. a partial non-understanding), she may choose to clarify those parts, by posing a request and thereby signal non-understanding in those concepts. S.9 in Table 1 is an example, where the system lacks confidence in the concept RIGHT. Such requests are often formulated as yes/no questions. If the addressee is missing some part of the utterance, the clarification request may instead be formulated as a wh-question. The system could, for example, have said:

(1) What do you have on your right?

Purver et al. (2001) explores the different forms that clarification requests may take, by studying the British National Corpus. An interesting finding is that 45% of the clarification requests were elliptical or fragmental (just like S.9 in Table 1). For concept-level error han-

dling, these are especially interesting, since they focus on problematic concepts and thereby make the dialogue more efficient.

It is important to note that while clarification requests signal non-understanding, they may also give positive evidence of understanding by display, as is illustrated in (1). In this example, the system displays that it has understood that the user has something on her right, but at the same time gives negative evidence on her understanding of what it is.

The most explored techniques for grounding in spoken dialogue systems are “explicit” and “implicit” verification, illustrated below:

(2) U.1 I have a large concrete building on my left
 S.2e Do you have a large concrete building on your left?
 S.2i (You have) a large concrete building on your left.
 S.2ii Which colour does the large concrete building that you have on your left have?

S.2e exemplifies explicit verification, which may be described as a clarification request. S.2i exemplifies implicit verification, where the system displays its understanding. These techniques may often be experienced as tedious and unnatural, since they operate on the utterance level and are realised as separate communicative acts. Implicit confirmation may also be integrated into the next act, as in S.2ii, but as Gabsdil (2003) notes, this is done on the utterance-level (including the whole previous utterance). As the example shows, this may sound unnatural and tedious, compared to just including the most important concepts, as in S.5.

Traum (1994) presents a computational model of grounding where a recursive transition network is used to model the stages of the grounding process, including acknowledgements, repairs, and request for repairs. Larsson (2002) describes a model for grounding utterances (or issues), where positive and negative evidence are given on all action levels, using the information state approach. While these models handle the process of grounding information, the units that are considered for grounding are utterances or issues. The examples provided do not show how individual concepts can be grounded. Another problem is that negative answers to grounding and clarification moves are not used as constraining information. In Larsson (2002), a “backup” copy of the dialogue state is kept to restore the state if the proposed interpretation is rejected. Consider example (2) above. If the user would answer “no”, the fact that the user does not have a large concrete building on her left is valuable information for constraining possible user positions. This shows that clarification requests should be treated using a general model for handling requests, including the support for negations.

Concept-level clarification requests have been studied to a greater extent than concept-level display. Rieser (2004) and Schlangen (2004) describe implementations of systems that are capable of posing fragmentary clarification requests based on concept confidence scores on all action levels. However, the models do not handle the user's reactions to those requests.

A dialogue system also needs to consider the case of complete non-understanding. Typically, the system says something like "Sorry, I didn't understand", thereby encouraging the user to repeat. Skantze (2005) shows in an experiment that this is not what humans typically do when faced with complete non-understanding. Instead, they tend to ask new task-related questions, without signalling non-understanding.

2.4 Late error detection

Clarification requests correspond to what Schegloff (1992) calls second-turn repair, i.e. the repair is done in the second turn counting from the problematic utterance. Third-turn repair occurs if an interlocutor give positive evidence of understanding and the first speaker realises that she has been misunderstood and initiates a repair. By displaying evidence of understanding, the system may detect errors by letting the user initiate a third-turn repair. *Late error detection* is the task of detecting that the user has initiated such a repair, i.e. to detect a previous misunderstanding.

Late error detection may also be performed if an interlocutor realises that her model of the world contains contradictory information. An example can be seen in Table 1. After turn U.4, the system believes that the user has a tree on her right (since there was an undetected misrecognition). But after turn U.10, the system realises that there is no place the user could be. The only fact that has a relatively low confidence and has not been grounded is that the user has a tree on her right. The system may now assume that this was a misunderstanding, and update its model.

2.5 Choosing error handling strategies

The previous discussion shows that there are several ways to handle uncertainty and errors in dialogue. A speaker may give evidence of understanding, request clarification on what is not understood, or presuppose understanding and defer the detection of errors to a later stage in the dialogue. As Allen et al. (1996) points out, sometimes it may be better to "choose a specific interpretation and run the risk of making a mistake as opposed to generating a clarification subdialogue". The choice of strategy should depend on the result of the early error detection, i.e. how confident the system is in its understanding, but also on the consequence that a misunderstanding would have; the cost of a potential misunderstanding should be compared to the cost of

making the grounding move. As an example, take utterance S.3 in Table 1. The system has a fairly high confidence in the user's position, but still chooses to give positive evidence of understanding, instead of deferring the detection. The reason is that if the system would misrecognise the user's goal, the system would not detect the error until the user has already reached the goal.

The possibility to defer error handling based on consequence of misunderstanding has not been explored to a great extent; confidence scores are most often only considered once and not stored for late error detection. To be able to defer error detection, as well as choosing from different error handling strategies on the concept level, the system should keep a model of when concepts have been grounded, by whom and how confident the system is in this.

3 The HIGGINS spoken dialogue system

In this section, the architecture, semantic structures and modules of the HIGGINS spoken dialogue system will be described.

3.1 Architecture

The HIGGINS spoken dialogue system is a distributed architecture with modules communicating over sockets. Each module has a clearly defined input and output (in XML), and can be implemented in any language, running on any platform. Figure 1 shows the most important modules and messages in HIGGINS, in the present configuration. The recognition result (the top hypothesis with word confidence scores) from the ASR is sent to an interpreter, called PICKERING. PICKERING recognises and creates semantic representations of communicative acts (CA:s) from the user (without considering the discourse context).

In HIGGINS, dialogue management is not implemented as a single module. Instead, this processing is divided into a *discourse modeller* (GALATEA) and an *action manager*. This division is similar to the approaches taken in Allen et al. (2001) and Pflieger et al. (2003). The communicative acts are sent from PICKERING to GALATEA, which does a context aware interpretation and builds a discourse model. The discourse model is then sent to the action manager, which consults the discourse model and the domain database to make decisions and send communicative system acts. These acts are sent back to the GALATEA, as well as to a generator. Thus, GALATEA models communicative acts both from the user and the system; ellipsis, anaphora and grounding status is handled and modelled in the same way for all communicative acts. The action manager may also make changes to the discourse model (if, for example, an error is detected) and send it back to GALATEA.

From the generator, the textual representation of the system’s communicative act (enriched with prosodic markup) is sent to a speech synthesiser (TTS).

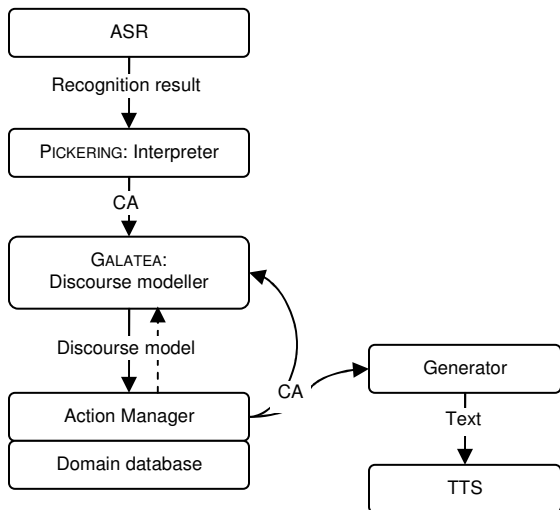


Figure 1: The most important modules and messages in the HIGGINS architecture. CA stands for communicative act.

GALATEA is fairly generic and can be configured with XML. The action manager, on the other hand, is highly domain dependent. However, much of the work that a typical dialogue manager has to do (such as ellipsis and anaphora resolution) is already resolved by GALATEA.

3.2 Semantic representations

Semantic descriptions are consistently represented as rooted unordered trees of semantic concepts. Nodes in the tree may represent for example attribute-value pairs, objects, relations and properties. Such structures are very flexible and can be used to represent deep semantic structures, such as nested feature structures, as well as simple forms, depending on the requirements of the domain. By using tree matching (similar to Kilpeläinen, 1992), a pattern tree can be used to search for instances in a given target tree. Thus, larger semantic structures can form databases which may be searched. It is also possible to include variables in a pattern tree for specifying constraints and extracting matching nodes, as well as using special pattern nodes for negation, etc.

The semantic tree structures in HIGGINS are represented with XML. Figure 2 shows an example, representing a wooden building, where a style-sheet (XSLT) has been used to visualise the XML as a graphical tree structure (using HTML). Values starting with a dollar sign are interpreted as variables. If this tree was used as a pattern in a database search, the result would be all possible bindings of variable ID4 (i.e. a list of the id:s of all the wooden buildings in the database). The database

in HIGGINS is a large XML document that contains all landmarks and their properties, as well as a set of possible user positions and how they relate to the landmarks.

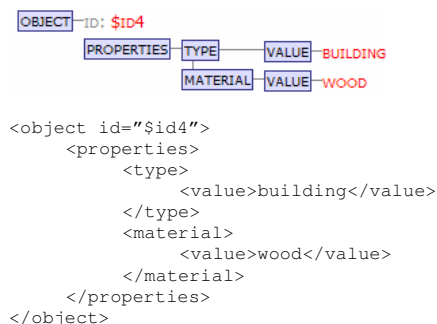


Figure 2: The semantic representation of a wooden building visualised graphically, as well as the corresponding XML.

The semantic representations may be enhanced with “meta-information”, such as confidence scores, communicative acts, and if information is new or given. Figure 3 shows the representation of the utterance “the building is made of wood”. The structure tells us that this is a communicative act (CA) of the type ASSERT, that the object is singular (SING), and that the object and type are GIVEN information but the material NEW. This meta-information can easily be removed to create a pattern like the one in Figure 2 for database searches (since such information is not contained in the database).

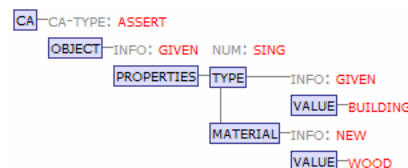


Figure 3: The semantic representation of the utterance “the building is made of wood”.

These structures make it fairly straightforward to represent the semantics of verbs, relations, etc. They are just represented as single nodes or tree fragments. It is not necessary to specify the arguments that for example a predicate must have. Tree structures may also be unified. A semantic template is used to specify how the nodes may be structured, to guide the unification. The template also makes it possible to unify structures starting at different levels in the tree.

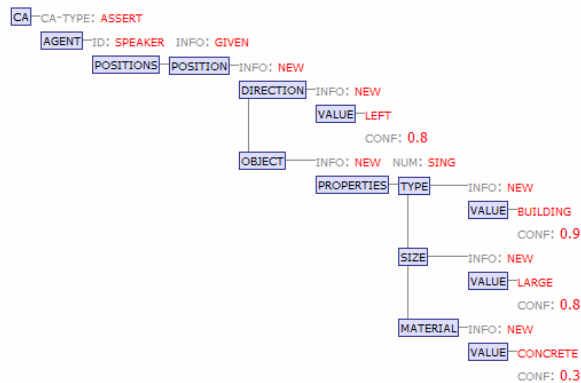
3.3 PICKERING: the semantic interpreter

The interpreter developed within HIGGINS is called PICKERING (Skantze & Edlund, 2004b) and is implemented in Oz². PICKERING uses a context free grammar

² <http://www.mozart-oz.org/>

to parse recognition results. The grammar is enhanced with rules for generating the kind of semantic trees described above. PICKERING can automatically make exceptions from the syntax given in the grammar by handling insertions and non-agreement inside phrases and by combining non-continuous phrases. While deviations from the grammar are allowed by PICKERING, they are taken into account when the scoring the interpretations. Since PICKERING has access to the semantic results, it can automatically filter out semantically equivalent solutions, by using tree comparison.

To make error handling on concept level possible, PICKERING also automatically transfers word confidence scores into the semantic trees. The semantic template used for unification can be marked with slots for confidence scores. The confidence scores for the words that are involved in creating a node with such a slot are then averaged to compute a confidence score for the node. When averaging the word confidence scores, they are weighted based on the length of the individual words, which gives a better result according to Gabsdil and Bos (2003). An example semantic result with concept confidence scores is shown in Figure 4



I HAVE LARGE HELLO CONCRETE BUILDING ON MY LEFT

Figure 4: The semantic result from Pickering, interpreting the first part of U.4 in Table 1, with concept confidence scores.

3.4 The action manager

The discourse model can be compared with the information state used in TrindiKit (Larsson, 2002). However, the discourse model only contains information about what has been said, not the system’s plans or agenda – this is modelled by the action manager. “Issues” are central concepts in the issue-based approach to dialogue management described by Larsson (2002). Issues are not directly modelled in GALATEA, since they are not needed for ellipsis resolution, anaphora resolution, or grounding tracking. Issues could be modelled in the

action manager, but they are not at the current moment. Instead, the action manager has a check-list that it goes through each time the discourse model gets updated. For example, after U.2 in Table 1, the system first checks if there are any concepts that should be grounded, and chooses to display understanding of “an ATM”. It then checks the discourse model and finds that the user’s goal is known. The next item on the check list is the user’s position, and since there is no information on that in the discourse model, the action manager poses an open request on the user’s position (S.3).

Since “issues” or “sub-dialogues” are not explicitly represented, the system does not need to know when issues are resolved or rejected. For example, if the system needs more information about the user’s position, it might ask “can you see a wooden building in front of you?”. If the user then says “I have a green building on my left”, the system may find out that it has enough information to continue with route directions. Whether the “issue” raised by the first question is resolved or not does not matter.

4 GALATEA: the discourse modeller

The discourse modeller developed within HIGGINS and introduced here is called GALATEA and is also implemented in Oz. The discourse modeller has three main tasks:

- Resolve **ellipses** by maintaining a **CA-list** (list of past communicative acts) in chronological order, with the most recent act first. Ellipses are transformed into full propositions, based on the CA-list.
- Resolve **anaphora** by maintaining an **entity list**, extracted from the CA:s, with the most recently mentioned entity first.
- When new CA:s are added to the model, **grounding status** is added to nodes in the semantic representation, i.e. information about who added the concept to the model, in which turn, and how confident the system is in the concept. This information is also transferred to the entity list.

The CA-list and the entity list form the discourse model, which is represented with XML.

4.1 Ellipsis resolution

GALATEA resolves ellipses by transforming them into full propositions. To do this, domain dependent *context rules* are used that transform communicative acts based on previous acts (similar to Carbonell, 1983). Table 2 exemplifies a transformation based on a rule that handles all answers to wh-requests (called content-requests here). The rule is applied when the new CA is an ellipsis,

and there is a wh-request in the CA-list with a requested node marked with `THEME:1`. If possible, a new CA of type `ASSERT` is created where the top node in the request is copied and the theme-node is unified with the first node that can be unified in the ellipsis (in this case the `COLOUR` node). If the unification fails, the rule is not applied.

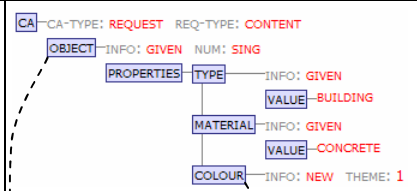
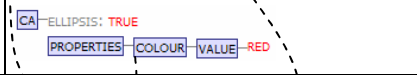
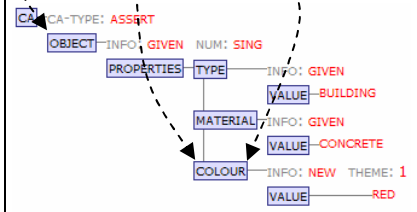
<p>Context: <i>Which colour does the concrete building have?</i></p>	
<p>Ellipsis: <i>Red</i></p>	
<p>Transformation: <i>The concrete building is red</i></p>	

Table 2: Example transformation.

The rules are written in XML, but will not be explained in detail here. The context may be a CA from the other speaker or the same. It is also, of course, possible to transform CA:s other than ellipses. Each rule has a fairly generic purpose. Currently, about 10 different context rules are used in HIGGINS.

4.2 Anaphora resolution

GALATEA has no access to the domain database. Thus, it cannot map entities (i.e. referents) in the discourse to real objects in the world. Instead, it keeps a list of entities that are mentioned (e.g. “a large building”) in the discourse and assigns variable id:s to them. For some entities (such as the user), an absolute id can be assigned. The action manager may then use the entities in the discourse model as patterns and make a database search to find possible referents, i.e. bindings to the entity id variables (as described in 3.2).

When semantic structures are created in PICKERING, they are marked with given/new status, based on definiteness and sentence structure. Some parts may be given and some new, for example when asserting information about a given object (see Figure 3 for an example). Each time an entity marked as new is added to the entity list, it is placed on top. If an entity marked as given is added (i.e. an anaphora), the entity list is searched from top to bottom for an antecedent. The nodes marked as given in the entity to be added are used as a search pattern and the potential antecedents as tar-

get, and a pattern match is performed. If an antecedent is found, it is moved to the first position in the entity list and unified with the added entity. If no antecedent is found, the added entity is treated as new and simply placed first.

Since assertions about entities are unified in the entity list, it is possible to refer to entities with descriptions they haven’t actually been directly referred with before. For example, there is a reference in utterance S.11, in Table 1, to “the red building”. There is no entity directly referred to in this way before, but the entity list will contain one after U.6.

The entity list may also be used by the action manager to select an appropriate referring expression (such as S.5 and S.11 in Table 1).

4.3 Grounding status

Handling anaphora and ellipses are necessary capabilities of a discourse modeller. In addition to this, GALATEA is also capable of tracking the grounding status of concepts. The grounding status is information about who added the concept to the model, in which turn and how confident the system is in the concept. Since the same concept may be mentioned several times, the grounding status is represented as a list of grounding data. This way, the system may model grounding information over time. This information may then be consulted for various error handling strategies, which is described in the next section. The grounding status can be compared with the “contextual functions” used in Heistercamp and McGlashan (1996), and the discourse pegs used in McTear et al. (2005), that are used to keep track of the system’s belief in what has been said.

The grounding information is added before resolving ellipses and anaphora. This ensures that only concepts that were part of the original utterance are grounded, not those that are added in the ellipsis resolution. In the semantic template used for unification, places where grounding information should be added are marked. An example of how grounding is updated is given in Table 3. As can be seen in the table, each `GROUNDING` tag contains a list of CA-tags, which represent communicative acts in which the concept was grounded. Each such tag contains information on the speaker (`AGENT`), the turn (`CAID`) and (if applicable) the concept confidence score (`CONF`), taken from the parse result (as exemplified in Figure 4).

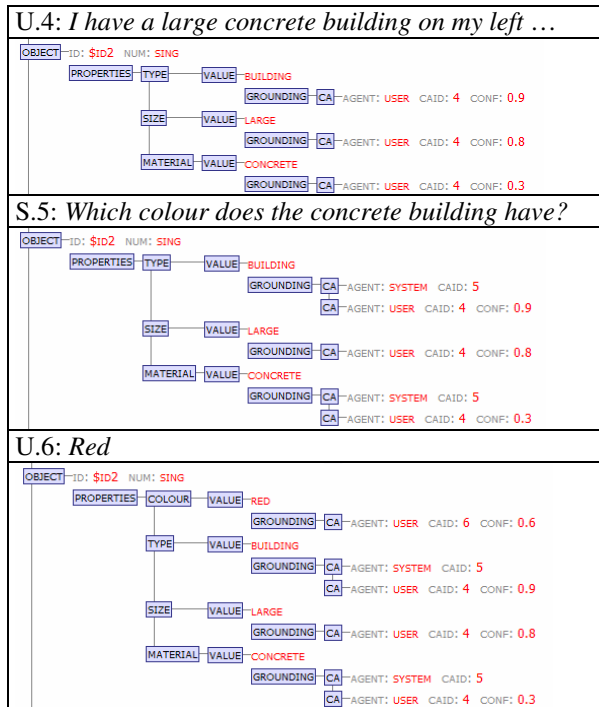


Table 3: How the grounding status for the entity that represents the concrete building gets updated.

5 Error handling strategies

The grounding status information in the discourse model may be used by the action manager to perform the various concept-level error handling strategies described in section 2.

Currently, a simple distinction is made between high and low grounding status. If a concept is only mentioned by the user with low confidence score, it has a low grounding status. If it has been mentioned by the system and/or by the user with a high confidence, it has a high grounding status. The threshold used for “high” and “low” confidence is different for different strategies.

5.1 Grounding

The entity list may be used by the action manager to display positive evidence of understanding, by searching for concepts with low grounding status. This may be done as a separate communicative act, as in S.3 in Table 1, or integrated in another act, as in S.5. Since GALATEA also models the system’s actions, those concepts will then have a high grounding status.

Every time the system refers to a given entity, appropriate integrated display of understanding is automatically done. To construct a referring expression to an entity that is on the entity list, the action manager simply makes a copy of the entity and removes all concepts

with high grounding status. This ensures that the concepts with low grounding status will get a high grounding status. An example is shown in Table 3. When the system needs to ask a question on the colour of the building, it copies the entity and removes the concept LARGE, since it has a high grounding status (based on the confidence score). The TYPE concept (BUILDING) is not removed, since it is needed for a valid referring expression (otherwise it would say “which colour does the concrete have”).

5.2 Clarification

The latest CA in the discourse model may be searched for concepts, or trees of concepts, with low grounding status. The action manager may then embed such a fragment in a CA of type request and send it, resulting in a fragmentary clarification request. When GALATEA receives this elliptical CA, it will be transformed into a full request. This way, the grounding status for the involved concepts will be updated correctly. Table 4 shows how a clarification dialogue with elliptical communicative acts is interpreted by GALATEA.

Turn	Utterance	After transformation
U.1	I have a red building on my left	[same]
S.2	red?	is the building red?
U.3	no	the building is not red
	green	the building is green

Table 4: How a clarification dialogue is interpreted by GALATEA.

Negations and confirmations are represented with POLARITY nodes that are attached to concepts. This makes it easy to represent and integrate “yes” and “no” answers, as well as adverbial negations. A concept may have several POLARITY nodes, which makes it possible for one participant to confirm something while another participant negates it. The POLARITY nodes are then taken into account when doing tree pattern matching. Figure 5 shows the resulting entity after the dialogue in Table 4. As can be seen, the negative answer is kept in the model, which is useful when constraining possible user locations.

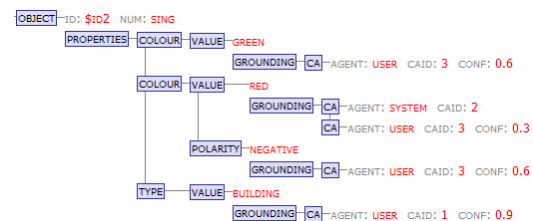


Figure 5: The resulting entity after the clarification dialogue.

5.3 Late error detection

Due to the misrecognition in U.4 (in Table 1), the discourse model will contain an error. However, after turn U.10, the system discovers that there is no place where the user can be. It may now use the grounding status in the discourse model to look for errors. The only concept with a low grounding status is shown in Figure 6. This concept may be removed, and the system will still have the information that the user can see a tree somewhere.

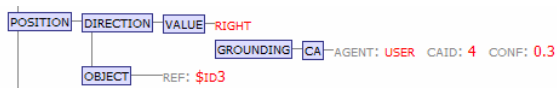


Figure 6: A concept with low grounding status is detected.

It is also possible to remove information that is associated with a specific turn (by looking at the CAID attribute in the grounding status). The most obvious case is when the system has just grounded some concepts and the user signals a problem. The concepts associated with the system's previous turn may then be removed. For example, if the user had signalled a problem after S.5 in Table 3, the system could remove the concepts CONCRETE and BUILDING, or add negative POLARITY nodes to them. The model would still contain the fact that the user has something large on her left.

Since the model also contains information about what the user has grounded, it is possible to detect cases where the user misunderstands the system. For example, the user never displays any understanding of the concepts WOOD and BUILDING in U.8 in Table 1, which can be detected in the discourse model.

5.4 Choosing strategy

Given these different error handling strategies the discourse model allows, the question is how to choose strategy. Generally, low confidence scores lead to clarification requests, mid confidence scores lead to display of understanding, and high confidence scores lead to no grounding actions. Since the choice of strategy is done in the action manager, it is possible to make a task-related choice, i.e. to have different confidence thresholds for different tasks, depending on the consequence of misunderstanding, as discussed in 2.5. For example, when the user asserts the goal, as in S.1 in Table 1, the system has a much higher threshold for not displaying understanding. When the user asserts her position in U.4, the system instead chooses to defer the handling of the low confidence score in the concept RIGHT.

6 Conclusions and future work

In this paper, the discourse modeller GALATEA has been presented. It models the grounding status of concepts mentioned during the course of the discourse by storing

confidence scores for individual concepts, together with information about when the concepts have been mentioned and by whom. It has been shown how this information may be used by an action manager for display of understanding, clarification requests, and late error detection, all on the concept level.

Currently, the thresholds used for different strategies and tasks have been manually tuned. Finding methods for automatic tuning is a topic for future research. The selection of clarification and grounding strategies is built into the same action manager as the one that handles the navigation task. It should be possible to build a separate generic action manager for handling these tasks, which could be reused in different applications.

The display of understanding and clarification requests described here currently only operates on the perception level on the action ladder described previously. In current spoken dialogue systems, this is probably where most errors occur, but a possible future direction is to extend the model to track confidence scores and grounding status for different levels. Schlangen (2004) presents a model where pragmatic confidence scores are used for clarification requests.

The next step is to conduct experiments with users to evaluate the system. An especially important topic is to look at how users give negative signals when the system gives positive evidence after misrecognitions, and how these negative signals may be detected, so that the errors in the discourse model may be corrected (as described in 5.3). Error recovery strategies after total non-understanding will also be studied.

Compared to full propositions, the meaning of an elliptical utterance is more dependent on prosodic features. We are currently investigating how prosodic features of (synthesised) elliptical clarification requests affect the perceived meaning (Edlund et al., 2005).

An important question is, of course, to what extent the techniques described in this paper may apply to other domains. GALATEA, as well as other HIGGINS components, is currently being tested in Connector, a dialogue system acting as an automatic switchboard and secretary. Connector is part of the EU-funded CHIL-project³ – a project investigating automatic tracking and support of interactions in meeting rooms.

Acknowledgements

This research was carried out at the Centre for Speech Technology, a competence centre at KTH, supported by VINNOVA (The Swedish Agency for Innovation Systems), KTH and participating Swedish companies and organizations. The author would like to thank the other participants in the HIGGINS project, Jens Edlund and Rolf Carlson, for valuable ideas and discussion.

³ <http://chil.server.de/>

References

- Allen, J. F., Miller, B. W., Ringger, E. K., and Sikorski, T. (1996). Robust understanding in a dialogue system. In *Proceedings of ACL '96*, 62-70.
- Allen, J. F., Ferguson, G., and Stent, A. (2001). An architecture for more realistic conversational systems. In *Proceedings of Intelligent User Interfaces 2001*.
- Anderson, A., Bader, M., Bard, E., Boyle, E., Doherty, G., Garrod, S., Isard, S., Kowtko, J., McAllister, J., Miller, J., Sotillo, C., Thompson, H., and Weinert, R. (1991). The HCRC Map Task corpus. *Language and Speech*, 34(4), 351-366.
- Baus, J., Kray, C., Krüger, A., and Wahlster, V. (2001). A resource-adaptive mobile navigation system. In *Proceedings of the International Workshop on Information Presentation and Natural Multimodal Dialog*.
- Carbonell, J. G. (1983). Discourse pragmatics and ellipsis resolution in task-oriented natural language interfaces. In *Proceedings of the 21st conference on Association for Computational Linguistics*.
- Clark, H. H. (1996). *Using language*. Cambridge: Cambridge University Press.
- Edlund, J., Skantze, G., and Carlson, R. (2004). Higgins - a spoken dialogue system for investigating error handling techniques. In *Proceedings of ICSLP 2004*, 229-231.
- Edlund, J., House, D., and Skantze, G. (2005). The effects of prosodic features on the interpretation of clarification ellipses. Submitted to *Interspeech 2005*.
- Evermann, G. and Woodland, P. C. (2000). Large vocabulary decoding and confidence estimation using word posterior probabilities. In *Proceeding of ICASSP 2000*, 2366-2369.
- Gabsdil, M. (2003). Clarification in spoken dialogue systems. In *Proceedings of the AAAI Spring Symposium on Natural Language Generation in Spoken and Written Dialogue*.
- Gabsdil, M. and Bos, J. (2003). Combining acoustic confidence scores with deep semantic analysis for clarification dialogues. In *Proceedings of IWCS-5*, 137-150.
- Gabsdil, M. and Lemon, O. (2004). Combining acoustic and pragmatic features to predict recognition performance in spoken dialogue systems. In *Proceedings of ACL*.
- Heisterkamp, P. and McGlashan, S. (1996). Units of dialogue management: an example. In *Proceedings of ICSLP 1996*, 200-203.
- Hirst, G., McRoy, S., Heeman, P., Edmonds, P., and Horton, D. (1994). Repairing conversational misunderstandings and non-understandings. *Speech Communication*, 15, 213-230.
- Kilpeläinen, P. (1992). *Tree matching problems with applications to structured text databases*. PhD Thesis, Department of Computer Science, University of Helsinki.
- Knight, S., Gorrell, G., Rayner, M., Milward, D., Koeling, R., and Lewin, I. (2001). Comparing grammar-based and robust approaches to speech understanding: a case study. In *Proceedings of Eurospeech 2001*.
- Larsson. (2002). *Issue-based dialogue management*. PhD thesis, Goteborg University.
- McTear, M., O'Neill, I., Hanna, P., and Liu, X. (2005). Handling errors and determining confirmation strategies - An object-based approach. *Speech Communication*, 45(3), 249-269.
- Pfleger, N., Engel, R., and Alexandersson, J. (2003). Robust multimodal discourse processing. In *Proceedings of DiaBruck '03*.
- Purver, M., Ginzburg, J., and Healey, P. (2001). On the means for clarification in dialogue. In *Proceedings of SIGdial 2001*.
- Rieser, V. (2004). *Confidence-based fragmentary clarification on several levels for robust dialogue systems*. MSc thesis, University of Edinburgh.
- Schegloff, E. A. (1992). Repair after next turn: the last structurally provided defense of intersubjectivity in conversation. *American Journal of Sociology*, 97(5), 1295-1345.
- Schlangen, D. (2004). Causes and strategies for requesting clarification in dialogue. In *Proceedings of SIGdial 2004*.
- Skantze, G. and Edlund, J. (2004a). Early error detection on word level. In *Proceedings of ITRW on Robustness Issues in Conversational Interaction 2004*.
- Skantze, G. and Edlund, J. (2004b). Robust interpretation in the Higgins spoken dialogue system. In *Proceedings of ITRW on Robustness Issues in Conversational Interaction 2004*.
- Skantze, G. (2005). Exploring human error recovery strategies: implications for spoken dialogue systems. *Speech Communication*, 45(3), 207-359.
- Traum, D. (1994). *A computational theory of grounding in natural language conversation*. PhD Thesis, University of Rochester.

Appendix A

The same dialogue as in Table 1. The fourth shows the interpretation of the utterance after ellipsis resolution. Important concepts are written in uppercase. Concepts that are grounded (i.e. displayed) in the same turn are also bold. The fifth column shows the error handling strategies that are used.

Turn	Transcription	ASR result	After ellipsis resolution	Error handling
S.1	Where do you want to go?		Where do you want to go?	
U.2	To an ATM	TO AN ATM	I want to go to an ATM	
S.3	Ok,		Ok	
	an ATM		You want to go to an ATM.	Separate display of understanding (ATM)
	Can you describe where you are now?		Can you describe where you are now?	
U.4	I have a large concrete building on my left	I HAVE LARGE HELLO CONCRETE BUILDING ON MY LEFT	I have a LARGE CONCRETE BUILDING on my LEFT	
	and a tree in front of me.	AND TREE ON RIGHT	I have a TREE in FRONT of me.	
S.5	Which colour does the concrete building have?		Which colour does the CONCRETE BUILDING have?	Integrated display of understanding (CONCRETE, BUILDING)
U.6	Red	RED	The CONCRETE BUILDING is red	
S.7	Ok,		Ok,	
	can you see a wooden building in front of you?		Can you see a WOODEN BUILDING in FRONT of you?	
U.8	No,	NO	I can not (NEGATIVE) see a WOODEN BUILDING in FRONT of me	
	but I have one on my right.	I HAVE ONE ON RIGHT	I have a WOODEN BUILDING on my RIGHT .	
S.9	On your right?		Do you have a WOODEN BUILDING on your RIGHT ?	Clarify RIGHT
U.10	Yes	YES	I do (POSITIVE) have a WOODEN BUILDING on my RIGHT .	
S.11	Ok,		Ok,	Late error detection of RIGHT in U.4
	I think I know where you are.		I think I know where you are.	
	Walk a little bit forward and take left after the red building.		Walk a little bit FORWARD and take LEFT AFTER the RED BUILDING .	

Appendix B

The complete entity list after turn U.10 in Table 1, before late error detection.

