Manual for the Analysis of Settlers Data

Nicholas Asher Vladimir Popescu Philippe Muller
Stergos Afantenos Anais Cadilhac Farah Benamara
Laure Vieu Pascal Denis

June 12, 2012

1 Introduction

Our project seeks to provide a multi-layered annotation of chat data obtained from on line play of the game *Settlers*. In *Settlers*, players are given positions on a territory with certain resources located in different areas. There are five resources: Sheep, Wheat, Clay, Wood and Ore (or Rock). The players then must build roads, settlements and cities to score points and to win the game.

The game is divided into rounds, where a given player, call him A, is in charge of the round, with the following structure:

1. A rolls the dice. The number determined by the dice will correspond to the number of one or more of the areas. Settlements and cities located next to that area or those areas can bring the owner of those developments additional resources.

2. If A needs more resources, she chooses one or more people to bargain with. Typically, players do not have enough resources of their own to win the game and so they must negotiate with other players to get the needed resources. A bargain involves an offer to trade a certain quantity of one of the resources for a quantity of another resource. Other players may accept the offer, reject it, or modify the offer by, say, changing the amount or type of resource to be traded. We call those modifications, *counteroffers*.

3. The players spend the resources they have after bargaining to build more developments.

A typical game session will involve dozens of such rounds. The game rules can be found at `http://www.catan.com/gamerules.html`.

We are interested in analyzing conversation among agents who have different and competing goals—what is known as noncooperative or strategic situations. The chat in *Settlers* is a way to observe such conversation in a non-experimental setting. We encourage you to look at the rules if you are interested in the game; although this is not necessary to do the annotation, an understanding of the rules may help you figure out what's going on in the chat.

Here are some things that we want to know and that conversational participants exploit to form ideas and plans (strategies) about what to say and what to do in the game.

- we want to know to whom a speaker is addressing his turn. It could be one or more people. If it's not clear, we want to know that too.
- we want to isolate strategic information, what trades were made, what trades were successful and what offers were unsuccessful
- revelation of preferences and at what level.
- comments on plays and strategies

We separate out this information into various levels of the annotation process.

The annotation has several levels.

1. Segmentation: First, there is the segmentation of the dialogue into separate turns. A turn contains what a player enters in the chat window at one time. Each turn has a speaker. The turns themselves are segmented into *elementary discourse units* or EDUs. The document you will receive will have each turn separated out, with the speaker marked, as well as the EDUs in each turn. EDUs within the turn are distinguished because each one carries a distinct role in the dialogue at some level of the annotation. For example, someone in a turn may acknowledge a previous turn, accept a proffered offer and explain why he or she is accepting the offer. We want to keep separate all of these bits of information. Many turns in our chat conversations, however, contain only one segment.
2. Discourse features.
   (a) The addressee: This level of the annotation is to determine for each EDU, whom is the speaker addressing. If it can't be determined, we will mark that with a '?'. If it is addressed to all the other players, we will mark 'All'.

(b) Type of speech act. This level has to do with the type of EDU. Is it a question, a request or an assertion.

(c) Dialogue acts relative to the game. The next level has to do with what is the function of the EDU with respect to the game. Is the speaker of the EDU making an offer? Is he or she making a counter offer, which may be a reply to an offer or a specification of an offer the speaker has already made? Is the speaker accepting or rejecting an offer? We also want to know whether the speaker is making a comment about what strategy some player (it could be the speaker herself) should make or has made—in which case, we will label the turn with *Strategic Comment*. There are also many other EDUs, which don't have a direct role with respect to the game; we label these *Other*.

(d) Rhetorical structure. A fourth level has to do with the rhetorical function the EDU plays. For example, a player may simply acknowledge the receipt of information in the previous term.

(1) a. That's the robber.
b. Right.

The response *right* acknowledges the information given by the previous speaker. There are many rhetorical functions that an EDU can play, which we detail below. These rhetorical functions are inherently relational. That is, one EDU plays a certain rhetorical function with respect to another discourse unit. We will ask you to link the EDU with the other unit or units and label the link. Sometimes several EDUs group together to form a common rhetorical function. This grouping is called a *complex discourse unit* or CDU. We give examples of this below.

(e) Preferences. The final level has to do with the preferences the speaker commits to if any. For instance, if the speaker says

(2) a. I'd like sheep.

we can infer that he clearly prefers to get sheep at the current state of the game.

2 Segmentation

The first task involved in building a discourse annotation for STAC is the identification of the elementary discourse units (EDUs) of a given text. EDUs are basic units that

combine together to form a coherent text. They are analogous to words that combine together to form a sentence. For the purposes of the current annotation campaign, the conversations are already segmented beforehand. However, for information purposes only, the conventions for segmentation are discussed in detail below. By convention, in the examples below, a & is placed after each EDU.

2.1 Base case

Segment boundaries are placed after all punctuation marking the end of a sentence, i.e., periods, question marks, and exclamation marks, if they exist. However, in chat the punctuation is somewhat arbitrary. All "full" clauses should be annotated, even when there are words missing or ellipsis. Here's an example of ellipsis.

(3) a. Dave: Do you have any wheat?
b. Tomm: Not & until someone starts rolling eights.

I have segmented *not* from the subordinate clause *until someone starts rolling eights* because there is an elided (main) clause associated with the negative particle: I don't have wheat.

2.2 Sentence internal segmentation

Subsentential constituents are treated as EDUS if they serve a discernible discourse function. In normal written text, punctuation and discourse markers are good surface syntactic cues for detecting sentence internal EDUs. Segment boundaries are placed *after* punctuation – including periods, commas, hyphens, colons and semi-colons – but *before* discourse connectors such as "and", "or", etc. and complementizers such as "that", "if", "whether", etc.

Also comment words like *sorry*, are to be segmented. Here is an example:

(4) a. Can you give me some wood for some sheep?
b. No.& I've only got one wood, &and I'm holding onto it. &Sorry

Three little dots can sometimes signal an ellipsis, especially if they signal a missing main clause. Consider:

(5) a. Now if John will just sit down, & ...

The three little dots . . . are an indication of an implicit main clause. So we have segmented it. Note that the . . . do not always indicate an ellipsis of a clause. In the next two examples the . . . don't indicate an elided clause, and so we won't segment.

(6) ....oh

(7) soo... how do we do this?

Some words or phrases like *Oh, OK, alright, right then* signal acknowledgements. Acknowledgments can function at different levels; you can say *OK* just to signal to your interlocutor that you have understood what he said. But you can also say "OK" if you understand and accept what he has said, say an offer. You can also acknowledge what someone has said, accept it, and be ready to move on to a new topic. Sometimes one turn may contain several acknowledgment markers. When these involve different levels, we segment them. For instances as in:

(8) a. Do you have any sheep?
b. No, I don't,& sorry.
c. OK,& Right then.

OK and *Right* in the last rejoinder involve two different acts of acknowledgement, one at a level of understanding, and one at a level of "I understand and accept what you have said, and I'm ready to move on".

The example above shows that we also segment out emotive or comment like phrases, like *sorry, great*. The turn

(9) Thanks,& sorry!

would have two segments, as would turns with emoticons.

(10) I have lots of sheep now& :)

2.2.1 Relative clauses

In the chat corpus, you may find a few relative clauses. Only nonrestrictive relative clauses introduce EDUs, unless doing so results in a discontinuous EDU. Restrictive relative clauses like

(11) Anyone who wants to can trade with the bank.

are never segmented, as they do not serve a discourse function, but rather restrict the denotation of a common noun. We have very few non restrictive clauses in our corpus

The following example is correctly segmented.

(12) I have two sheep,& which gives me my settlement.

Here's another:

(13) a. You can trade for more sheep with Rennoc,
b. who's drowning in sheep.

Clauses Clauses introduced by subordinating conjunctions such as *when*, *while*, *until* etc., are assumed to be EDUs. This follows the event rule stated above, as such conjunctions generally take tensed clauses as arguments. All tensed clauses are assumed to introduce an eventuality.

(14) Also Dave, if you keep too many cards, & the robber can take them away.

Notice that we are not segmenting out the explicit addressee. This information is important but is not a separate EDU.

Here are some more examples involving adjunct clauses:

(15) a. We'll steal& when we're past diplomacy& :P
b. if you have another ore& i can do it again
c. Or if you want& we could do that now too

Infinitival VPs that function as purpose clauses are treated as separate discourse units. Purpose clauses are identified using the *in order to* test: replace *to* with *in order to* and

check for semantic equivalence.

For example:

(16) He wants ore& to build a city.& Then he'll win.

Adverbials The segmentation of adverbials, especially adverbial prepositional phrases (PPS), is tricky. Luckily, adverbial PPs occur rarely in our chat corpus, and when they are they are often hiding an elliptical clause. A

(17) in time, and with enough 8s& i am sure I'll be well wooded

The adverbial *with enough 8s* is short for the antecedent of a conditional *if I get enough 8s*.

2.2.2 Coordination

Occasionally speakers in the chat will use coordinated constructions. When there's an explicit discourse function involved, as in the example below, we will segment these constructions.

(18) I'm first gonna trade with Dave& and then with you, Rennoc.

All coordinated tensed clauses function as discourse units. Nonclausal coordinate structures, for instance VPs, sometimes, but not always, introduce EDUs. Coordinated VPs are treated as separate discourse segments when they either include a discourse particle or contain discourse structure within (at least one of) the coordinated constituents.

We can also have coordinations between questions, as in

(19) is Euan being conspicuously silent& or looking elsewhere?& :D

Here are some examples where we segment because two offers are being proposed.

(20) a. I am willing to do wheat –> clay,& or wheat for ore
b. anyone want wheat for sheep?& or wheat for ore?
c. would offer 1 wood for 1 sheep & / 1 wood for 1 wheat

Note that non standard symbols (–>) for prepositions and (/) for 'or' are sometimes used.

On the other hand, as you will see below, we have a separate annotation for offers, statements about having resources in which we can put certain Boolean values. So we would not segment:

(21) a. I dont have sheep or wood Euan
b. anyone want wheat or wood for sheep?
c. i have a clay or a wood& i'm happy to dispense for wheat

Here (21)c has, technically, an appositive or non restrictive relative clause *i'm happy to dispense for wheat* that is linked to the complex noun phrase or determiner phrase (DP) *a clay or a wood*.

In (21)a we have a Has resources speech act which tells us that the speaker has neither sheep nor wood. In (21)b,c we have offers where the speaker is offering either of two resources for one that he wants. We do not segment these since we encode this information directly into the offer.

3 Features of Discourse Constituents

3.1 Addressee

Typically each EDU has an emitter, the person who is speaking; this is provided with the annotation. There is also the intended recipient, the person to whom the speech act is addressed. It may be addressed to the whole group of players or to any subset thereof or to another individual. Sometimes it may not be clear who's being addressed, in which case this field should be completed with a '?'.

3.2 Surface form or surface speech act

Each EDU has a form; it's either a question, a request, or an assertion. Here is an example of an assertion:

(22) I need sheep.

Here's an example of a question:

(23) Do you want to trade?

And here's an example of a request:

(24) Give me two sheep.

3.3 Dialogue acts relative to the game

Each EDU may also have a role in the bargaining component of the game. Thus, we will label EDUs as OFFER, COUNTEROFFER, ACCEPT (OFFER), REFUSAL (OFFER), STRATEGIC-COMMENT, or OTHER. With many of these types, we will associate features.

Offers
Let's first look at OFFERs. OFFERs can be expressed by questions, requests or assertions:

(25) Tomm: Dave, will you trade 1 wheat for 1 ore?

(26) Tomm: Dave, give me 1 wheat for 1 ore.

(27) Tomm: Dave, I will trade 1ore for 1 wheat.

In this offer, we notice an offer recipient who is typically the addressee of the turn (in this case Dave), an offerer (the speaker in this case Tomm), some resources offered (in this case 1 ore), and some resources requested (in this case 1 wheat). We can think of a complete offer then as coming with four "slots" that need to be filled. The way we will implement this in the annotation is to first annotate to whom the offer is addressed, the addressee, and then to determine in the sentence what resources are give-able or offered and what resources are receivable or wanted. In the examples above, the word 'ore' denotes a resource whose status is give-able, whose kind is ORE, and the quantity of which is 1. The word 'wheat' denotes a resource whose status is receivable, whose kind is WHEAT and the quantity of which is 1. You can think of a complete offer as always coming with four such features, which we arrange in the form of a list below.

Offer maker: Tomm, Offer receiver: Dave, Resource (Status: give-able,

kind: ORE, quantity: 1), Resource (Status: receivable, kind: WHEAT, quantity: 1)

Many of the offers made in the text are not complete offers. For instance,

(28) Dave: Does anyone have any wheat?

would count as an offer but with the addressee either specified as 'All' or as unspecified '?'. Furthermore, the sentence only mentions one resource 'wheat'. The status of this resource is give-able, and its kind is WHEAT, but the quantity of the give-able resource is unspecified.

Offer maker: Dave, Offer receiver: All or ?, Resource (Status: receivable, kind: WHEAT, quantity: ?)

In some offers the resources offered or requested can be complex. For example,

(29) Cardlinger: Does anyone have any wheat or clay?

This counts as an underspecified offer with the feature structure:

Offer maker: Cardlinger, offer receiver: All, Resource (Status: receivable, Kind: WHEAT, quantity: ?) OR Resource (Status: receivable, Kind: CLAY, quantity: ?)

Note that the resources talked about in the sentence have just one status; they are receivable but they are of several types. The annotation tool will allow us to conjoin or disjoin resources giving us in effect here a disjunctive kind WHEAT OR CLAY for the receivable resource. The way we implemented this was to have the annotator specify each resource mentioned and then if need be group them into conjunctions or disjunctions of resources. In this case the EDU states an offer about receivable resources of either the WHEAT or CLAY kind.

There are other ways of stating resources that look complex.

(30) Verena: I will give wheat for anything except sheep.

In this example what is special is the type of resource that Verena wants. We would annotate the resource denoted by 'sheep' as

> Resource (Status: receivable, kind: anything but WHEAT, quantity: ?)

The full offer looks like this:

> Offer maker: Verena, Offer receiver: All or ?, Resource (Status: give-able, kind: WHEAT, quantity: ?), Resource (Status: receivable, kind: anything but WHEAT, quantity: ?)

Counteroffers

Let's now look at the type COUNTEROFFER. Counteroffers are offers that are responses to an offer.

(31) a. Tomm: Dave, will you trade 1 ore for 1 wheat ?
b. Dave: How about 2 wheat for 1 ore?

Dave is making a counteroffer to Tomm's offer. He's changing the quantity of wheat that he wants to receive in exchange for giving away an ore.

Like OFFERS, COUNTEROFFERS also have an addressee or recipient and an offer maker. They also typically involve resources. In this case, we have:

> Offer maker: Dave, offer receiver: Tomm, Resource (Status: receivable, Kind: WHEAT, quantity: 2), Resource (Status: give-able, Kind: ORE, quantity: 1)

Counteroffers, like Offers, can be incomplete and can leave one of the 4 slots above uninstantiated.

Another feature common to Counteroffers is that the resources they specify may be *anaphoric* or of a special nature.

(32) a. Tomm: Dave, will you trade 1 ore for 1 wheat ?
b. Dave: How about 2 for 1 ?

In ((32)b) Dave is referring to a trade in which he gets 2 wheat for 1 ore. But here the resources 'wheat' and 'ore' have to be retrieved from the offer to which Dave is

replying. In this case, we would annotate the numerals '2' and '1' with the Resources feature, but set the kind to ANAPHORIC. The annotation tool will then allow you to specify what kinds of resources are being referred to in these cases.

Sometimes a player will give a counteroffer to his own offer:

(33) a. A: Does anyone want sheep for ore?
b. A: Does anyone want 2 sheep for an ore?.

A's second question is a counteroffer to his first offer, which may have no takers.

ACCEPT and REFUSAL
We now come to ACCEPT and REFUSAL. These are related to OFFERS or COUNTEROFFERS and are largely self-explanatory. Let's look at a continuation of (31):

(34) a. Tomm: Dave, will you trade 1 ore for 1 wheat ?
b. Dave: How about 2 wheat for 1 ore?
c. Tomm: No, not interested& Sorry

Here Tomm refuses Dave's counteroffer. Here's an example of an acceptance:

(35) a. Joel: Do you want a clay for a wheat
b. Euan: I can wheat for clay.
Here Euan accepts Joel's offer. We typically put what is being accepted or rejected as an argument of ACCEPT or REFUSAL. We note that often a refusal or an acceptance can convey information about what resources are give-able or receivable.

An EDU can perform an act of commenting on a move or strategic option. We label this as STRATEGIC-COMMENT.

(36) Dave: Nice roll

Here Dave comments on someone's roll of the dice; he approves presumably because it gives him resources that he needs.

Often speakers comment on moves by others. In the following TOMM steals some of Dave's resources and Dave replies:

(37) Really!

This is a strategic comment that shows Dave's disapproval.

More important are explicit mentions of strategies.

(38) Rennoc: Listen to me Dave,& don't trade with tomm.& or he'll win.

The middle EDU in the example above is a STRATEGIC-COMMENT; it's a request to Dave not to trade with Tomm.

Other The final category of EDU with respect to domain level acts is OTHER. This concerns EDUs that do not have any relevance to the game.

(39) Euan: And I alt-tab back from the tutorial& What's up?

Here Euan is explaining why he hasn't been paying attention to what's going on: he's been looking at the game's on-line tutorial. There are many such moves. Many comments concerning trades often given with emoticons are labelled OTHER.

(40) a. A: Do you have any sheep?
b. B: No& Sorry

B's comment "Sorry" is not really a strategic comment; it's not about the game but really just a mark of politeness. We label this with OTHER.

In many cases, an ACCEPT, REFUSAL, STRATEGIC-COMMENT or OTHER also conveys information about resources an agent has, even though this information is not intrinsic to these linguistic actions in the way that it is to OFFERS and COUNTEROFFERS. For these actions, we can sometimes specify resources. For example:

(41) a. Tomm: Dave, will you trade an ore for a wheat ?
b. Dave: I only have wood sadly.

In ((41)b) Dave is refusing Tomm's offer. But he's also saying something about his resources. He's saying that he has wood. In this case we would annotate ((41)b) as a REFUSAL but we would also label the resource denoted by 'wood' as

Resources(status: Possessed, Kind: WHEAT, quantity: ?)

We can also have ACCEPT acts like the following, which conveys the possession of certain relevant resources:

(42) a. A: Does anyone have any wheat or clay?
b. B: Yes.

In this case, B's turn is labeled with ACCEPT. If A's question is an OFFER, by answering "yes" to it, B also conveys the information that he has some of the requested resources. Hence, we don't need to annotate any resource unit in B's answer. Moreover, he doesn't explicitly mention any resource.

Wheat OR Clay.

Here's an example of a REFUSAL that conveys a lack of the relevant resources:

(43) a. Joel: hey anyone have any brick?
b. Cardlinger: Nope.

Cardlinger's reply "Nope" is a REFUSAL of Joel's offer, but it also conveys that he has 0 clay. In this case, however, we don't need to annotate Cardlinger's reply with a resource unit since we can infer that he has 0 clay from his answering the question (something we'll come to in the next section), and indeed we can't because he doesn't mention any resource.

4 Discourse Structure

4.1 Discourse relations

The game sessions we have are composed of a series of bargaining dialogues, each one demarcated by the roll of the dice by one of the players. That is, a roll of the dice will precede each bargaining session and a roll of the dice will follow each bargaining session. We're interested in the relations between discourse units whether they are EDUs or CDUs within one of these bargaining dialogues, although there may be relations between discourse units belonging to different bargaining sessions. So, before you start annotating discourse relations, check to see whether the discourse unit you are trying to relate is at the beginning of one of the bargaining sessions. If it is, it *may* relate to a discourse unit in the previous session, but it may not. It may just be an initial discourse unit, to which subsequent discourse units are related.

Between EDUs as well as between CDUs, or between CDUs and EDUs of a bargaining session, there are discourse relations that relate one EDU or CDU to another EDU or CDU and characterize the discourse role of the EDU or CDU in the text. Ideally, every discourse unit within a bargaining session should be related to some other unit in the session. Theories of discourse structure countenance many such relations, but we will only be using a few here.

Continuation	Narration
Result	Correction
Elaboration	Explanation
Conditional	Alternation
Contrast	Parallel
Answer / Question answer pair	Commentary
Q-elab / Follow-up question	Clarification Q
Acknowledgement	Background

Elaboration $Elaboration(\alpha, \beta)$ relates EDUs or CDUs whenever the second unit provides more information about the eventuality introduced in the first constituent. For example, $Elaboration$ holds if the main eventuality of the second EDU is a sub-type or part of the eventuality mentioned in the first. As such, a relation of temporal inclusion often holds between the related eventualities. A discourse marker such as *for instance*, or the explicit listing of sub-events (*first*, *second*, etc.), are good cues to $Elaboration$.

In our corpus, elaborations typically occur, when an agent makes an offer and then further specifies it. For example:

(44) a. A: Does anyone have any wheat?
b. A: I have ore

In (44)b the speaker is specifying his offer given in (44)a. You should annotate this discourse relation with $Elaboration$((44)a, (44)b).

Sometimes speakers will follow up one question with another. These questions often stand in an elaboration like relation. Here is an example:

(45) a. A: Does anyone want sheep for ore?

b. A: Does anyone want 2 sheep for 1 ore?.

Explanation $Explanation(\alpha, \beta)$ holds when β explains why, or gives the cause of, what happened in α. *Because* is an important cue for $Explanation$ (see (46)). But it doesn't always occur when we have an explanation (see (47)).

(46) Dave: I only have one,& and I'm holding onto it, & because I want to build a road.

(47) Dave: I don't want any sheep.& I'm already drowning in sheep.

The third constituent of (46) explains why the speaker is holding on to the card that he doesn't want to trade. The second clause of (47) explains why the speaker doesn't want any sheep, even though there's no "because" or "since" that would be an explicit marker for this relation.

Sometimes, explanations go across turns. This explanation goes across two turns by the same speaker.

(48) a. Tomm: Rennoc had lost about half way through to be honest
b. Tomm: I sort of cut him out

We would annotate this example with $Explanation$((48)a, (48)b).

Acknowledgement These are very common in the chat corpus. They are signaled by words like *OK, Right, Right then, Good, Fine, etc.*. It's often difficult to determine whether the acknowledgment signals an understanding of what was said, an acceptance of what was said or an acceptance and a signal to change the topic of conversation or move on. It's also often difficult to determine what is being acknowledged.

(49) a. A: Does anyone have any ore?
b. A: I'm offering sheep.
c. B: OK.

Here it's hard to tell what sort of acknowledgment *OK* is signaling. But if we look at a next segment, it sometimes becomes clear.

(50) a. A: Does anyone have any ore?
b. A: I'm offering sheep.
c. B: OK.
d. B: How about 1 ore for 2 sheep?

(50)d is a counter offer of the offer made in (50)a and (50)b. This indicates that the acknowledgment in (50)c is really an acceptance of what was said and of the offer made in (50)a and (50)b. Usually, an acknowledgement is linked at least to the previous segment; sometimes, its first argument may include more segments, as in (50). Sometimes, it's hard to tell exactly what the first argument of the $Acknowledgement$ relation is. So in the annotation tool below, you will see that we have made it possible to say that you aren't sure of what the scope or the first argument of the acknowledgement is.

Question Answer Pair This relation is used to link an answer to the question it is an answer to. This comes up a lot in our corpus, as many offers are in the form of questions:

(51) a. A: Does anyone want sheep for ore?
b. B: No.

The relation between (51)a and (51)b should be annotated as QAP((51)a, (51)b).

Follow-Up Question Sometimes a follow-up question is intended to get more information in order to answer a first question. The follow up question will lead to an answer to the first question. Here is an example.

(52) a. A: Does anyone have sheep?
b. B: Do you have any clay?

Presumably if B gets an answer to his question he will reply to A's question and offer of trading something for sheep. This should be annotated $Q - elab$((52)a, (52)b).

Here's a different type of follow up question, where the follow up question attaches to an assertion.

(53) a. Dave: did you just lose four resources?
b. Rennoc: yes
c. Dave: how?

d. Rennoc: rolled a 7.

We would annotate this example as QAP((53)a,(53)b), $Q-elab$((53)b,(53)c), and QAP((53)c, (53)d).

Clarification Question Clarification questions can occur at different levels. One is at the domain level. Here Rennoc didn't follow what happened. So (54)a is a clarification question.

(54) a. Rennoc: What just happened?
b. Dave: Tomm just stole some of your resources.

Clarification questions can also occur concerning what was said. In the following example, Dave isn't clear what Rennoc was asking?

(55) a. Rennoc: Does anyone have sheep or rock?
b. Dave: Sheep or ore?
c. Rennoc: Yes

Comment $Comment(\alpha, \beta)$ holds if β provides an opinion or evaluation of the content associated with α.

Surface cues for $Commentary$ include speaker-oriented adverbs, such as the supplemental uses of "luckily", "amazingly", etc, and utterance modifiers such "frankly", "confidentially", etc. Our corpus is also full of markers for comment, like *sorry*, *Ooh*, *bah*, *bollocks*, etc. Emoticons also indicate Comments.

(56) a. No.
b. Sorry

Sorry is a typical comment marker. We have put the two segments in (56)a,b. We would annotate the link between the two segments as $Comment$((56)a, (56)b).

Narration $Narration(\alpha, \beta)$ holds when the main eventualities of the EDUs α and β occur in sequence.

(57) a. I'm first gonna trade with Dave

b. and then with you, Rennoc.

Here the speaker sequences his planned trades. We annotate this with $Narration$((57)a, (57)b).

Continuation $Continuation(\alpha, \beta)$ is like $Narration$, except there is no sequence of actions. $Continuation$ often holds between two EDUS when they both elaborate or provide background to the same segment.

(58) a. A: Does anyone want to trade?
b. A: I'm offering sheep,& I want ore.

The two EDUs in (58)b. are linked by Continuation. Together, they form a complex discourse unit which elaborates on A's trade offer (58)a. More specifically, denoting by (58)b.1 the first segment of (58)b. "I'm offering sheep," and by (58)b.2 the second segment of this turn "I want ore.", the rhetorical structure of (58) is annotated with $Elaboration$((58)a., $Continuation$((58)b.1, (58)b.2)) or as:

$Elaboration$((58)a., [(58)b.1, (58)b.2])

$Continuation$((58)b.1, (58)b.2).

Contrast $Contrast(\alpha, \beta)$ holds when α and β have similar semantic structures, but contrasting themes, i.e. sentence topics, or when one constituent negates a default consequence of the other. ***But, however, on the other hand, nevertheless*** are all strong cues for $Contrast$. Postposed ***while***-clauses also sometimes introduce $Contrast$.

(59) a. I have sheep
b. but I can't trade
c. because it's Rennoc's turn.

This example is annotated with $Contrast$((59)a, (59)b) and $Explanation$((59)b, (59)c).

Parallel $Parallel(\alpha, \beta)$ has the same structural requirements as $Contrast$, but instead requires α and β to share a common theme. Cue phrases such as ***too*** and ***also*** are good indicators of $Parallel$. Here's an example

(60) a. You can trade for more sheep with Rennoc,
b. who's drowning in sheep.
c. You can also trade with me
d. :)

We have $Parallel$((60)a, (60)c). Note that (60)b explains why you can trade with Rennoc, so we would also annotate this example with $Explanation$((60)a,(60)b). The final emoticon is a comment, that might attach to (60)c but also might attach to the complex segment consisting of (60)a, (60)b, and (60)c. We'll talk about complex segments in the next section.

Note that in our corpus $Parallel$ and $Contrast$ may typically occur with ellipsis:

(61) a. Euan: Anyone have any sheep?
b. Joel: Nope
c. Jon: Same here

((61)b) links with Parallel to ((61)c), even though the structure of the two clauses is on the surface rather different. Important clues to the Parallel relation in this case are the fact that we have the word "same" and also an ellipsis, which forces a parallel semantic structure at a deeper level.

Result $Result$ connects a cause to its effect, i.e., the main eventuality of the first argument is understood to cause the eventuality given by the second. *So* is a good marker for $Result$. Anytime you can happily introduce a *So* between two discourse units, the relation that holds between them is probably $Result$.

(62) a. You only have 1 road segment.
b. So you can't build a settlement

We annotate this example with $Result$((62)a, (62)b).

Here's an example without an explicit marker, where we can happily add a *So*

(63) a. Cardlinger: I'm after wheat,& I'm afraid.
b. Joel: I have wheat.
c. Joel: We can talk about it on the next turn.

Here a result of Joel's having wheat is that he will talk about a trade on the next turn. We annotate this example with $Result$((63)b, (63)c). A more subtle use of $Result$ occurs between ((63)a, (63)b). Here the causal link is between Cardlinger's expressing a desire for wheat, and Joel's assertion that he has wheat. Cardlinger's expression of his wish results in Joel's stating that he has something that can grant Cardlinger's wishes.

Background $Background(\alpha, \beta)$ holds when β provides some stage setting for the event that happens in α. Sometimes we have $Background(\beta, \alpha)$ where the background comes first, as in (64), where the second segment sets the stage for the first one:

(64) Dave: While we're talking about trading,& does anyone have sheep?

Denoting by (64)1 the segment "While we're talking about trading," and by (64)2 the segment "does anyone have sheep", this example is annotated as $Background$((64)1, (64)2).

Conditional and Alternation $Conditional$ and $Alternation$ correspond rather closely to logical operations. $Conditional$ marks the presence of a conditional, while $Alternation$ marks the presence of a disjunction between two clauses. $Conditional$ is normally introduced by an *if...then* and so the first discourse unit is a hypothesis while the second is a consequence of the hypothesis. $Alternation$ is almost always introduced by *or*.

The clauses in the following example are related by $Conditional$:

(65) a. if you have another ore
b. i can do it again

Here are examples of $Alternation$:

(66) a. I am willing to do wheat –> clay,
b. or wheat for ore

(67) a. anyone want wheat for sheep?
b. or wheat for ore?

The example below is also treated as $Alternation$.

(68) a. I'm keeping my sheep
b. unless you can give me ore.

Note that $Alternation$ holds here between clauses not between noun phrases within an offer. Remember that offers can have complex Boolean arguments (like *sheep or ore*).

4.2 Complex Segments

Sometimes an argument to a rhetorical relation can involve more than one EDU. This often happens with offers.

(69) a. Does anyone have any ore?
b. I'm offering sheep.
c. OK.
d. How about 1 ore for 2 sheep?

The acknowledgment in (69)c carries over the whole offer in (69)a and (69)b. So we would annotate this example with $Elaboration$((69)a, (69)b),
$Acknowledgement$([(69)a, (69)b], (69)c), $Q-elab$((69)c, (69)d) and
$Q-elab$([(69)a, (69)b], (69)d).

Let's return to this example:

(70) a. You can trade for more sheep with Rennoc,
b. who's drowning in sheep.
c. You can also trade with me
d. :)

The final emoticon is a comment, that might attach to (70)c but also might attach to the complex segment consisting of (70)a, (70)b, and (70)c. In the latter case, we have a relation between one EDU and a complex segment, which we would annotate as follows: $Comment$([(70)a, (70)b, (70)c], (70)d).

5 Annotating Conversations in the Glozz tool

5.1 Overview

Each conversation associated with a Settlers game session is annotated in the Glozz tool, in terms of **units** and, when applicable, **relations** between units. The annotation is performed according to a Glozz annotation scheme which thoroughly follows the elements presented in the previous sections of this manual.

For each conversation, a Glozz pre-annotation is made available beforehand to the annotators. Such a pre-annotation contains information pertaining to the way the linguistic exchanges fit into the Settlers game session.

5.1.1 "Dialogue" units

Each conversation is already split into "Dialogue" units; each "Dialogue" unit contains a set of "Turns", delimited by a roll of the dice by the player in charge of the round.

Each "Dialogue" unit has a feature structure associated to it; this feature structure contains three features:

1. `Trades` – the resources (and their quantities) exchanged between pairs of players, if applicable; it has the form: "rennoc1 traded 2 sheep, 1 wood for 1 clay from Dave.";
2. `Dice_rolling` – the dice rolling of each player, if applicable; it has the form: "Dave rolled a 1 and a 5";
3. `Gets` – the resources that each player gets, along with their quantities; it has the form: "rennoc1 gets 1 sheep".

5.1.2 "Turn" units

The structural constituent of each "Dialogue" unit is the "Turn" unit, i.e. a chat line contribution of one player. Several "Turn" units make up a "Dialogue".

Each "Turn" unit is pre-annotated with a feature structure comprising the following attributes:

1. `Emitter` – the player having produced the "Turn", e.g. 'Euan';
2. `Identifier` – a numeric label which uniquely identifies the "Turn" in a conversation;
3. `Timestamp` – the moment in time when the "Turn" was produced, up to milliseconds precision;
4. `Resources` – the quantities of the resources (viz. clay, ore, sheep, wheat and wood) that the `Emitter` possesses when producing the "Turn";
5. `Developments` – the (non-zero) number of roads, settlements and cities already built by the `Emitter` by the time s/he produces the "Turn";
6. `Comments` – a free field where the annotator can add her/his comments on the "Turn" unit.

5.1.3 "Segment" units and their customizations

Each "Turn" unit is built of "Segment" units, that is, Elementary Discourse Units (EDUs). The borders of the "Segment" units are pre-annotated in that EDUs are already available when starting to annotate the conversations.

The "Segment" units have no features associated with them. However, the annotators will change the type of the "Segment" units, by customizing them according to their type of task-level dialogue act, viz. "Has_resources", "Offer", "Counteroffer", "Accept", "Refusal", "Strategic_comment", and "Other" units.

Each such customized "Segment" unit has a feature structure with two entries:

1. `Addressee` – the set of players that the "Segment" is addressed to; this set can be unspecified / unknown;
2. `Surface_act` – the type of the utterance through which the "Segment" is conveyed; can be either 'Question', 'Request' or 'Assertion';

5.1.4 "Complex discourse unit" schemata

The annotator can group several "Segment" units into "Complex_discourse_unit" schemata, in order to deal with the situation when several "Segment" units are in the scope of a rhetorical relation (see below). "Complex_discourse_unit" schemata do not have any feature.

5.1.5 "Resource" units

Inside each customized "Segment" unit, i.e. "Offer", "Counteroffer", "Other" etc. (see above) we can have "Resource" units, which identify resource-denoting words, viz. 'clay', 'ore', 'sheep', 'wood' and 'wheat'.

Unlike the "Segment" units which are already created beforehand when starting the annotations, the "Resource" units have to be created by the annotators themselves. Hence, whereas for "Segment" units all the annotators need to do is customize them (by changing their label into e.g. "Offer", "Counteroffer", "Other" etc.) and select the appropriate feature values for them, for "Resource" units, the annotators must first create them when needed, then select the appropriate feature values for them.

Each such "Resource" unit has four features:

1. `Status` – it can be either one of 'Possessed', 'Not possessed', 'Givable', 'Not givable', 'Receivable' and 'Not receivable'; note that 'Givable' implies 'Possessed', but not the other way around; likewise, 'Not possessed' implies 'Not givable', but not the other way around;

2. `Quantity` – a number of items, from 0 to 9; it can also be unspecified ('?');

3. `Correctness` – specified whether the resource is correctly stated ('True') or not ('False'); it can be unspecified ('?');

4. `Kind` – specified one of the five resource kinds stated above, as well as 'Anything but', plus one of the five resource kinds stated above, e.g. 'Anything but clay', which means that the resource involved is anything other than clay. We also have 'Nothing', for segments like "I don't have anything", where 'anything' is a "Resource" unit. We also have 'Anaphoric' when the resource is expressed through a plural anaphoric pronoun – e.g. "these", "each", "those" etc which refers to several resources stated in other turns.

5.1.6 "Several_resources" schemata

Two "Resource" units can be grouped in a "Several_resources" schema when two "Resource" units are in an "Offer" unit and have the same value for the `Status` attribute.

Each such "Several_resources" schema has one feature, `Operator`, which can be either 'AND' or 'OR', corresponding to the way of combining the two resources. When several resources are involved instead of two, the grouping in the "Several_resources"

schemata is recursively applied, i.e. at first, the two leftmost "Resource" units are grouped in a "Several_resources" schema, then this schema is grouped with the third "Resource" unit in a new "Several_resources" schema, and so on.

5.1.7 More on "Resource" units

In some cases resources can be referred to by plural pronouns such as "these", "those", "each", etc. Consider the following example:

(71) a. Euan: Anyone got clay or wood for wheat?
b. Jon: I can do that - 1 for 1 ?
c. Euan: Sure.
d. Euan: Or 2 wheat for 1 of each.

In (71)d., "each" refers to the "clay" and "wood" resources, which are 'Receivable' for Euan. However, "clay" and "wood" are both the 'Receivable' resources in Euan's offer (71)a. Therefore, "each" should be annotated as a "Resource" unit of a special kind, 'Anaphoric', with the same status as the resources it refers to (in this case, 'Receivable'), and of an unspecified quantity ('?').

If the plural pronoun refers to a set of resources of different statuses, then the status of the plural pronoun will be left unspecified ('?'), as in the following example, where "ore" is 'Givable', whereas "wheat" is 'Receivable' and hence "these" will have the status set to '?':

(72) a. A: I can give you ore for wheat.
b. B: I don't have any of these.

In both cases of plural anaphoric pronouns, we also need to specify what the pronoun refers to. Hence, one will use a special relation, *Anaphora*, which will link the pronoun to each of the resources it refers to or, when the resources are part of a "Several_resources" scheme, the *Anaphora* relation will link the pronoun to that scheme. Hence, "each" in (71)d. will be linked via one *Anaphora* relation to the "Several_resources" scheme which contains "clay" and "wood". In (72)b., "these" will be linked via two *Anaphora* relations to "ore" and "wheat" respectively. Each such *Anaphora* relation will go from the pronoun to each of the resources the pronoun refers to.

Occasionally, resources can also be expressed via elliptical constructions, when only the quantity of the resource is specified. In this case, this quantity should be annotated as a "Resource" unit in the usual way. The disambiguation of the elliptical constructions stems from the appropriate value of the Kind feature. To better see this, let us see how we annotate the two occurrences of "1" in (71)b.

The first occurrence of "1" refers to "clay or wood". Hence it is both an anaphora and an ellipsis. In other words, "1" should be annotated as being of 'Anaphoric' kind, quantity '1', status 'Receivable'. The second occurrence of "1" refers to "wheat" only. Hence, it should be annotated as being of 'wheat' kind, quantity '1', status 'Givable'.

In case of elliptical constructions or of singular anaphoric pronouns (e.g. "it"), the *Anaphora* relation need not be used, since the link is implicitly specified by properly setting the Kind attribute to the type of resource the anaphoric pronoun or the quantity in an elliptical construction refers to. Hence, in this case, the "Kind" feature will **not** be set to 'Anaphoric'.

Occasionally the resources may be described by something like "anything but wheat", in which case the `Quantity` feature will be set to '?' (i.e. the amount of resources is unspecified); the `Kind` feature will be set to e.g. 'Anything but wheat', as in the example:

(73) a. Amanda: ore?
b. Amanda: or anything really except wheat?

5.1.8 "Preference" units

Inside each customized "Segment" unit, i.e. "Offer", "Counteroffer", "Other" etc. (see above) we can have "Preference" units, which identify verbs that express preferences, e.g. "prefer", "like", etc.

"Preference" units have no features.

Unlike the "Segment" units which are already created beforehand when starting the annotations, the "Preference" units have to be created by the annotators themselves. Hence, whereas for "Segment" units all the annotators need to do is customize them (by changing their label into e.g. "Offer", "Counteroffer", "Other" etc.) and select the appropriate feature values for them, for "Preference" units, the annotators must first create them when needed.

5.1.9 Discourse relations

"Segment" units (EDUs) can be connected via discourse relations. Discourse relations can also connect "Complex_discourse_unit" schemata (CDUs), or even EDUs with CDUs.

Each discourse relation thus has two arguments, which can be EDUs or CDUs. The order of the arguments is important.

Each discourse relation must be of one of the 16 types specified above in this manual, and has two features:

1. `Comments` – a free field where the annotator can add her/his comments on the discourse relation;
2. `Argument_scope` – specified or unspecified, depending on whether both arguments of the rhetorical relation are known or not, respectively.

A synthetic view of these annotation elements is provided in Table 1.

5.1.10 Commented annotation example

Before dwelling on the technical details of the annotation tool (Glozz) used for annotating Settlers conversations, we will present, in a detailed manner, how a typical conversation excerpt is being annotated[1].

(74)
a. A: Hey anyone have any brick?
b. A: Or clay,& sorry.
c. B: nope& :(
d. C: no
e. A: I'm willing to do wheat for clay or ore.
f. D: And I alt tab back from the tutorial.& What's up?
g. A: do you want to trade?
h. B: A fancies a bit your clay.
i. A: yes!
j. D: watcha got?
k. A: wheat and wood.
l. D: Great.& I can do 1 of each for 2 clay.

[1]For presentation reasons, the example is an adaptation of a real conversation occurring in the context of a Settlers game.

Item	Type	Attribute	Value
Unit	Dialogue	Trades	who traded which resources with whom
		Dice_rolling	who rolled which values
		Gets	who got which resources
	Turn	Emitter	who produced the turn
		Identifier	unique ID of the turn
		Timestamp	time of the turn
		Resources	resource_type = quantity
		Developments	devel_type = number
		Comments	free comments
	Segment	—	—
	[one of the following: Offer, Counteroffer, Accept, Refusal, Strategic_comment Other]	Addressee	for whom the EDU is
		Surface_act	Question / Request / Assertion
	Resource	Status	Possessed / Not possessed / Givable / Not givable / Receivable / Not receivable / Anaphoric / ?
		Quantity	? / 0-9
		Correctness	True / False / ?
		Kind	clay / sheep / ore / wheat / wood / Anything but {sheep / ore / wheat / wood} / Nothing / Anaphoric
	Preference	—	—
Schema	Complex_discourse_unit	—	—
	Several_resources	Operator	AND / OR
Relation	[one of the 16 discourse relations] + $Anaphora$	Comments	free comments
		Argument_scope	Specified / Unspecified

Table 1: Glozz annotations units for the STAC project

EDU-level annotation

First, we discuss the EDU-level annotation.

▷ In (74)a., A asks for clay (actually, "brick" is a word for clay). By doing so, he makes an underspecified "Offer" dialogue act. The offer is underspecified because A only states what he wants, not what he is willing to give in return. So, A's turn (74)a. consists of a single EDU of type "Offer", with the following features: `Addressee` set to 'All', `Surface_act` set to 'Request'. Inside this EDU, a unit of type 'Resource' is created, containing only "brick". The feature structure of this resource unit is: `Status`: 'Receivable'; `Kind`: 'clay'; `Quantity`: '?'; `Correctness`: 'True'.

▷ Then, A's turn (74)b. contains two EDUs: the annotation of the first one is identical to the annotation of the EDU in (74)a.. The second EDU in this turn (what follows after '&') is labelled as "Other", with the same `Addressee` as before and with `Surface_act` set to 'Assertion'.

▷ Then, B's turn (74)c. consists of two EDUs. The first EDU is labelled as "Refusal" and annotated with the following features: `Addressee` set to 'A'; `Surface_act` set to 'Assertion'.The second EDU is labelled as "Other" and has the same `Addressee` and `Surface_act` feature values as the first EDU. In this case, this second EDU can be left unannotated, because it only contains an emoticon, which is not a strategic comment.

▷ C's turn (74)d. has exactly the same annotation as the first EDU in (74)c.

▷ A's turn (74)e. contains a single EDU which is labelled as "Offer" with the following features: `Addressee` set to 'All'; `Surface_act` set to 'Assertion' (or perhaps 'Request' ?). Inside this offer we create three 'Resource' units, for "wheat", "clay" and "ore". The "clay" 'Resource' unit has the following features: `Status` set to 'Givable'; `Kind` set to 'wheat'; `Quantity` set to '?'; `Correctness` set to 'True'. The "clay" and "ore" 'Resource' units both have `Status` set to 'Receivable', `Quantity` set to '?' and `Correctness` set to 'True'. However, the "clay" 'Resource' unit has `Kind` set to 'clay', while the "ore" 'Resource' unit has `Kind` set to 'ore'. Moreover, it this turn, we also have a "Several_resources" schema which contains the "clay" and "ore" resources and has its `Operator` feature set to 'OR'.

▷ D's turn (74)f. consists of two EDUs, both of type "Other" and with the same feature values for `Addressee` set to 'All'.The only difference consists in having `Surface_act` set to 'Assertion' for the first EDU and to 'Question' for the second EDU.

▷ A's question (74)g. consists of a single EDU which is labelled as "Offer". This offer is underspecified, because neither the givable nor the receivable resources are specified.

Its feature values are Addresee set to 'D' and Surface_act set to 'Question'.

▷ B's turn (74)h. consists of a single EDU labelled as "Strategic_comment", because it tells D something about possible goals of A. The feature values for this EDU are: Addressee set to 'D'; Surface_act set to 'Assertion'.Then, a 'Resource' unit containing "clay" is created, with the following feature values: Kind set to 'clay', Quantity set to '?', Status set to 'Receivable', and Correctness set to 'True'. This EDU also contains a "Preference" unit consisting of "fancies". This is to highlight the presence of a preference inside this EDU.

▷ A's turn (74)i. consists of a single EDU labelled as "Other", with Addressee set to 'B, D' and Surface_act set to 'Assertion'.

▷ D's turn (74)j. consists of a single EDU labelled as "Counteroffer". It is an underspecified offer, because no resource is stated. Its feature values are just Addressee set to 'A' and Surface_act set to 'Question' (because D is just asking A what he has).

▷ A's turn (74)k. consists of a single EDU labelled as "Strategic_comment", since A states what he has, perhaps implicating what he is willing to give to D. The feature values of this EDU are: Addressee set to 'D', Surface_act set to 'Assertion'.Inside this EDU two 'Resource' units are created, for "wheat" and "wood". Their feature values are: Status set to 'Possessed' (we don't know for sure that A is willing to actually give these resources in exchange for something else; all he is saying is that he possesses the resources), Kind set to 'wheat' for the first 'Resource' unit and to 'wood' for the second 'Resource' unit, Quantity set to '?' and Correctness set to 'True'. Moreover, we create a "Several_resources" schema containing both "wheat" and "wood" 'Resource' units. Its Operator feature is set to 'AND'.

▷ D's turn (74)l. consists of two EDUs. The first one is labelled as "Accept" and has the following feature values: Addressee set to A, Surface_act set to 'Assertion'.The second EDU is labelled as "Counteroffer" and has the same Addressee and Surface_act values as the first EDU. Inside this second EDU we create two 'Resource' units, one for "each" and one for "clay". The 'Resource' unit for "each" has the following feature values: Status set to 'Receivable', Kind set to 'Anaphoric', Quantity set to '2' (because of "1" in "1 of each" and of the fact that "each" points to a combination of two kinds of resources – "wheat and wood"), Correctness set to "True". Now, to mark that "each" refers to "wheat and wood" in A's turn (74)k., we create an *Anaphora* relation from the "each" 'Resource' unit to the "Several_resources" schema in A's turn (74)k. The 'Resource' unit for "clay" has the following feature values: Status set to 'Givable', Kind set to 'clay', Quantity set to '2', Correctness set to "True".

Discourse-level annotation

Second, we discuss the discourse-level annotation. Unless stated otherwise, all the relations will have their `Argument_scope` set to specified, since in most of the cases we can confidently pinpoint the arguments of the relations.

▷ By stating the first EDU in (74)b., A corrects himself by replacing "brick" with the commonly accepted word for this resource – "clay"; hence, we create a $Correction$ relation between these two turns. Hence, if we denote by (74)b.-1 the first EDU in A's second turn, we have: $Correction$((74)a., (74)b-1). Furthermore, by stating his second EDU (74)b.-2 ("sorry"), A just comments on his correction. We will thus create a $Comment$ relation between these two EDUs: $Comment$((74)b.-1, (74)b.-2).

▷ (74)c. and (74) d. are both answers to A's offer, which consists in the complex discourse unit [(74)a., (74)b.-1]. Hence, we create two Question-answer pair (QAP) relations: QAP((74)c., [(74)a., (74)b.-1]) and QAP((74)d., [(74)a., (74)b.-1]).

▷ In (74)e., A elaborates on his previous offer (i.e. the complex discourse unit [(74)a., (74)b.-1]). Hence, we annotate $Elaboration$([(74)a., (74)b.-1], (74)e.).

▷ In (74)f., D just enters the conversation, with no strong connection to what has being said previously. In this case, we will just put a $Continuation$ from the previous turn (74)e. and D's first EDU, (74)f.-1 ("And I alt tab back from the tutorial."): $Continuation$((74)e., (74)f.-1). In this particular case, given that we don't know precisely to which previous EDU (74)f.-1 is linked, we will set `Argument_scope` to 'Unspecified'. Then, in (74)f.-2, D asks "What's up" as a result of him being occuppied with alt-tabbing from the tutorial. We will hence annotate $Result$((74)f.-1, (74)f.-2).

▷ In (74)g., A asks D a follow-up question on whether he wants to trade or not. This question just follows D's previous question (74)f.-2. Hence, we have $Q-Elab$((74)f.-2, (74)g.).

▷ Then, B barges in in (74)h., by explaining D why A had asked him if he wanted to trade. We thus have an $Explanation$((74)g., (74)h.).

▷ A acknowledges this in (74)i. Hence, we have $Acknowledgement$((74)h., (74)i.).

▷ In (74)j., D asks A a follow-up question. It can be related either to to A's question (74)g., or to the complex segment [(74)g., (74)h., (74)i.]. If we choose the latter, then we have $Q-Elab$([(74)g., (74)h., (74)i.], (74)j.).

▷ In (74)k., A answers D's question (74)j. Hence, we have QAP((74)j., (74)k.).

▷ In (74)l.-1, D acknowledges A's answer, hence $Acknowledgement$((74)k., (74)l.-1). Then, D elaborates on his acknowledgement: $Elaboration$((74)l.-1, (74)l.-2).

Synthetically, the EDU- and discourse-level annotations of this conversation excerpt looks like in Table 2[2].

5.2 The Glozz annotation tool

The conversations for the games are annotated with the Glozz tool. The tool can be best run on any Windows or Linux machine with at least 1 GB of RAM memory and with the latest Java Runtime Environment installed.

With this tool, the annotators are able to:

1. load the annotation files, containing pre-annotations;
2. add their own annotations, thus updating the annotation files;
3. save the updated annotation files.

Loading the files needed for the annotation

Once the Glozz interface is open, it looks like in Figure 1. We observe a toolbar menu (A) in the upper side of the main Glozz window, along with three main vertical areas.

The leftmost one (B) shows an overall view of the text to be annotated. The area in the middle (C) shows the text to be annotated. Turns are marked in light gray, while elementary discourse units (segments) are marked with dark gray.

The rightmost area (D) contains four fields: a toolbar on the upper side; a frame where annotation units and relations can be chosen; a frame with feature-value pairs, where annotators can specify the feature values; finally, a "command-line" window, where annotations are represented in an internal format, which can also be used by the annotators. However, the usage and handling of the commands in this window is out of the scope of this manual.

The steps to load the files needed for the annotation are described below and illustrated in Figure 2.

[2]In this table, the following acronyms have been used: "Pref." for "Preference"; "Surf._act" for "Surface_act"; "Qty" for "Quantity"; "Req" for "Request"; "Receiv" for "Receivable"; "Poss" for "Possessed"; "Assert" for "Assertion"; "Ques" for "Question"; "Spec" for "Specified"; "Unspec" for "Unspecified"; "Cntoffer" for "Counteroffer"; "Str_comm" for "Strategic_comment"; "Elab" for "Elaboration"; "Cont" for "Continuation"; "Explan" for "Explanation"; "Ackn" for "Acknowledgement".

Turn			Dialogue act			Resource[a]				Rhetorical relation		Pref.
Id	**Utterance**	**Emitter**	**Type**	**Addressee**	**Surf._act**	**Operator[b]**	**Status**	**Kind**	**Qty**	**Type**	**Argument_scope**	
1	Hey anyone have any brick?	A	Offer	All	Req	–	Receiv	clay	?	–	–	–
2	Or clay,& sorry.	A	Offer	All	Req	–	Receiv	clay	?	*Correction*	Spec (1, 2-1)	–
			Other	All	Assert	–	–	–	–	*Comment*	Spec (2-1, 2-2)	–
3	nope& :(	B	Refusal	A	Assert	–	–	–	–	*QAP*	Spec (1, 3-1)	–
			Other	A	Assert	–	–	–	–	*Comment*	Spec (3-1, 3-2)	–
4	no	C	Refusal	A	Assert	–	–	–	–	*QAP*	Spec (1, 4)	–
5	I'm willing to do wheat for clay or ore.	A	Offer	All	Assert	OR	Receiv	clay	?	*Elab*	Spec ([1, 2-1], 5)	–
							Receiv	ore	?			
						–	Givable	wheat	?			
6	And I alt tab back from the tutorial.& What's up?	D	Other	All	Assert	–	–	–	–	*Cont*	Unspec (5?, 6-1)	–
			Other	All	Ques	–	–	–	–	*Result*	Spec (6-1, 6-2)	–
7	do you want to trade?	A	Offer	D	Ques	–	–	–	–	$Q-Elab$	Spec (6-2, 7)	–
8	A fancies a bit your clay.	B	Str_comm	D	Assert	–	Receiv	clay	?	*Explan*	Spec (7, 8)	✓
9	yes!	A	Other	B	Assert	–	–	–	–	*Ackn*	Spec (8, 9)	–
10	watcha got?	D	Cntoffer	A	Ques	–	–	–	–	$Q-Elab$	Spec ([7-9], 10)	–
11	wheat and wood.	A	Str_comm	D	Assert	AND	Poss	wheat	?	*QAP*	Spec (10, 11)	–
							Poss	wood	?			
12	Great.& I can do 1 of each for 2 clay.	D	Accept	A	Assert	–	–	–	–	*Ackn*	Spec (11, 12-1)	–
			Cntoffer	A	Assert	–	Receiv	Anaphoric	2	*Elab*	Spec (12-1, 12-2)	–
						–	Givable	clay	2			

Table 2: EDU-level annotations for example (74).

[a] Since the `Correctness` feature of the "Resource" units is always set to 'True', we chose to omit it from the table.

[b] This is the feature of "Several_resource" schemata, when applicable.

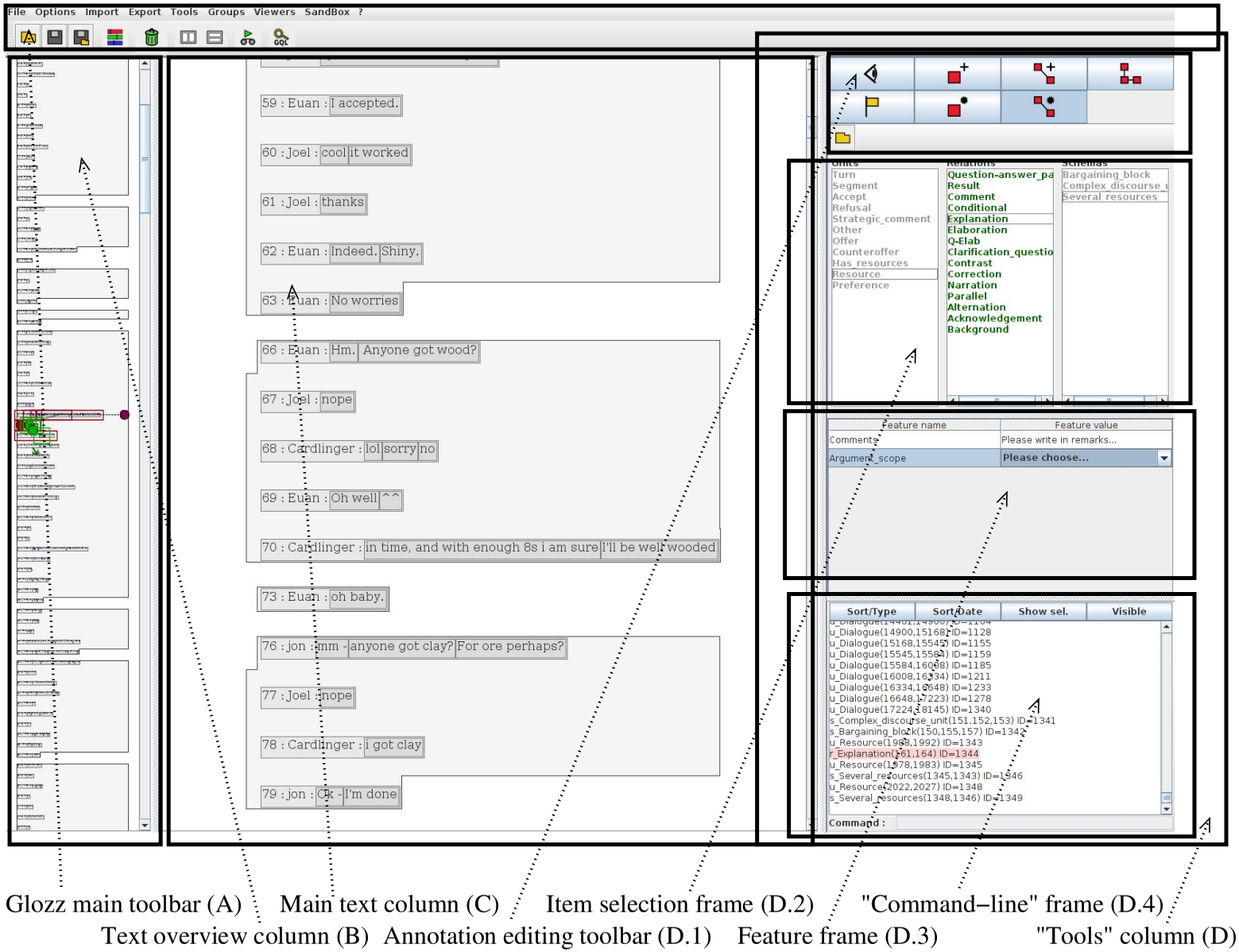

Figure 1: Overview of the Glozz interface

Figure 2: Loading a document into Glozz

1. Load the document to annotate, by clicking on the button  , situated on the toolbar. Two items have to be loaded:

   - the text (file with .ac extension)
   - its annotations (file with .aa extension).

   You can also load a file which contains annotations that you had previously performed and saved. The two files must have the same name, except for their extensions, e.g., annotation_1.ac and annotation_1.aa.

2. Load the annotation model, by clicking on the button  in the Annotation model area, to the right of the text zone. Please choose the file that bares the same name as the annotation files, except that it has the .aam extension (e.g. annotation_1.aam). The different elements of the annotation model thus appear in the Annotation model area.

3. Load the stylesheet, which allows one to color the elements of the annotation model. Click on the button  , situated on the toolbar, then the button  (`Open style`) in the foreground window. Please choose the `Stac.aas` file. You can alter the display colors as you wish, by clicking on the colored straps situated in the `Background-color` column.

Annotating units

Now that the text is open in the annotation interface, we can see that a pre-annotation is already present. It concerns the pre-annotation of "Dialogue", "Turn" and "Segment" units and the initialization of the following attributes of the "Dialogue" units: `Trades`, `Dice_rolling` and `Gets`, and of the "Turn" units: `Identifier`, `Emitter`, `Timestamp`, `Resources` and `Developments`, as discussed previously.

At this stage, you can customize and annotate the "Segment" units by changing them from "Segment" to either one of "Offer", "Counteroffer", "Accept", "Refusal", "Has_resources", "Strategic_comment", "Other". To this end, you should:

1. left-click, in the Annotation editing toolbar (the D.1 area on Figure 1), on the button  . Please pay attention to the presence of the $*$ on this button and of the "Edit units" legend.

2. left-click on the **Segment** unit that you wish to annotate, then left-click, in the Item selection frame (the D.2 area on Figure 1), on the appropriate unit, e.g. "Offer", "Other", etc.

3. perform the annotation, by clicking on the value of each feature (in the "Feature Value" column in the Feature frame – the D.3 area on Figure 1) and choosing the appropriate value, according to Table 5.1.9.

In order to annotate other "Segment" units, one must repeat steps 2 and 3 for each "Segment" unit. In order to be able to see and, if needed, edit the feature structure for each unit, one has to perform step 1 and then left-click on each unit.

This setup is illustrated in Figure 3.

In order to create "Complex_discourse_unit" schemata, please refer to the **Annotating schemata** subsection, below.

In order to annotate "Resource" and "Preference" units, the following steps should be observed:

1. left-click, in the Annotation editing toolbar (the D.1 area on Figure 1), on the button  . Please pay attention to the presence of the + on this button and of the "Add unit" legend.
2. left-click on the left frontier of the word(s) to be annotated (e.g. 'clay' for "Resource" units, or 'would rather' for "Preference" units), then on the right frontier of the word(s); thus, a new unit is created;
3. for editing the features associated with the "Resource" units, left-click, in the Annotation editing toolbar (the D.1 area on Figure 1), on the button  .

Annotating discourse relations

In order to add a discourse relation:

1. left-click on the button  on the Annotation editing toolbar (D.1 on Figure 1).
2. left-click on one of the 16 relation labels in the `Relations` column in the Item selection frame (D.2 on Figure 1). Please make sure that the arrow which corresponds to the relation is selected before choosing the relation label.
3. left-click on each of the "Segment" units and / or the "Complex_discourse_unit" schemata: first, on the left argument of the relation; second, on the right argument of the relation.

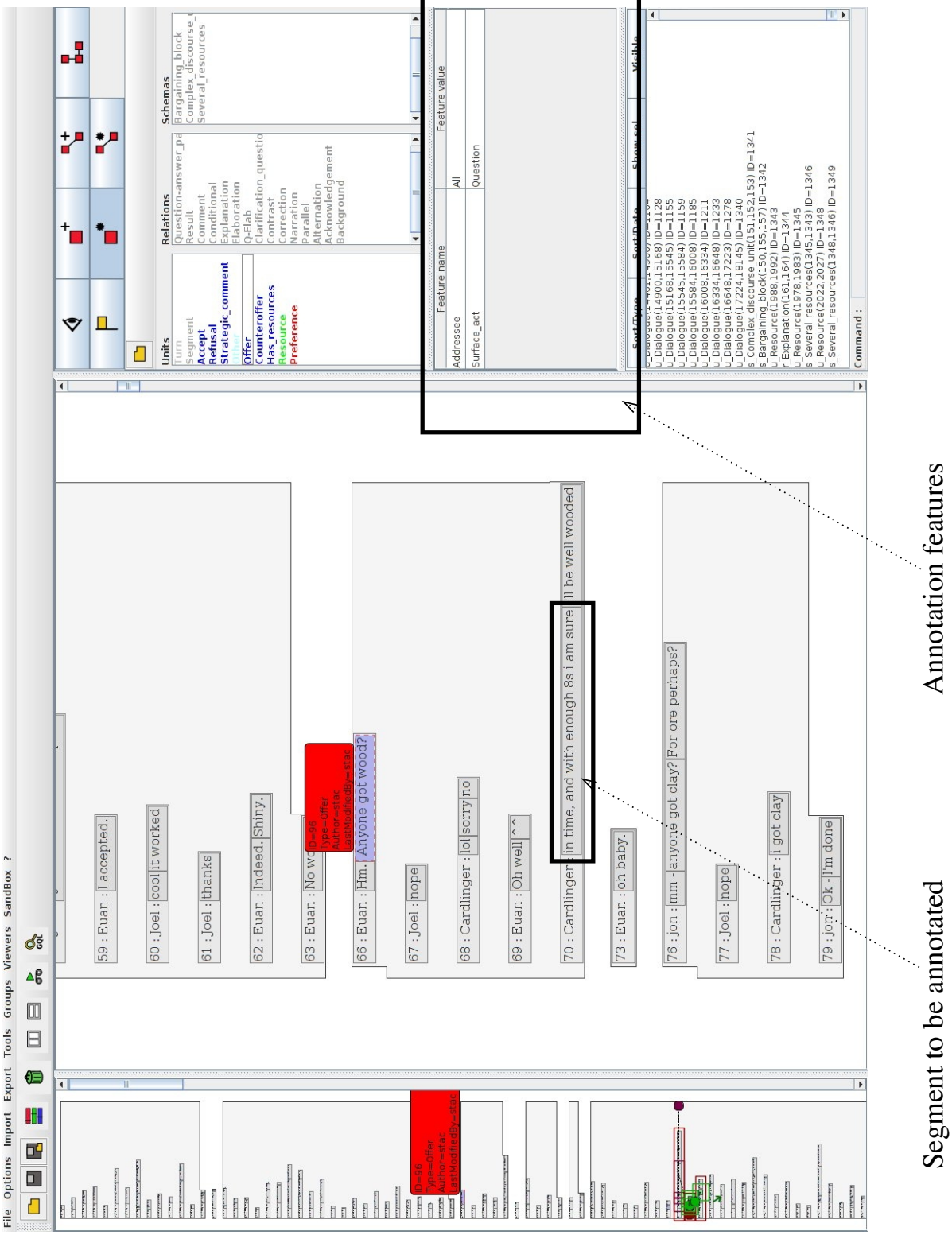

Figure 3: Editing a segment annotation in Glozz

In order to add new discourse relations, repeating steps 2 and 3 for each pair of arguments is sufficient.

In order to change the label of a discourse relation and / or to annotate its features:

1. left-click on the button  in the Editing Zone. Please pay attention to the asterisk $*$ on this button.
2. left-click on the arrow which corresponds to the relation
3. if necessary, left-click on the new relation label in the `Relations` column in the Item selection frame (D.2 on Figure 1). Please make sure that the arrow which corresponds to the relation is selected before choosing the relation label. Then, select the appropriate value for the `Argument_scope` feature: 'Specified' if both arguments are identified in full confidence; 'Unspecified' if one hesitates for at least one of the arguments, e.g. when having $Acknowledgement$ relations, where the left argument is sometimes hard to determine.

In order to edit other relations, repeating steps 2 and 3 for each pair of arguments is sufficient. Please pay attention to the fact that discourse relations link "Segment" units and / or "Complex_discourse_unit" schemata only. A "Turn" unit cannot be an argument of a rhetorical relation.

Relation editing is illustrated in Figure 4.

Annotating schemata

Once the units have been annotated, you may group, where appropriate, several units into a schema. Depending on the type of units, three kinds of schemata can be created:

1. several (customized) "Segment" units are an argument of a discourse relation; in that case, a "Complex_discourse_unit" schema should be created, so that it contains these "Segment" units. Please pay attention to the fact that any "Complex_discourse_unit" schema remains inside a "Turn" unit.
2. several "Resource" units are inside an "Offer" or "Counteroffer" unit and have the same value of the `Status` feature, i.e. either 'Givable' or 'Receivable'; in that case, a "Several_resources" schema should be created so that it contains the first two resources; then, if needed, a new "Several_resources" schema is created, so that it contains the first "Several_resources" schema, along with the third resource, and so on; subsequently, the `Operator` features of all the "Several_resources"

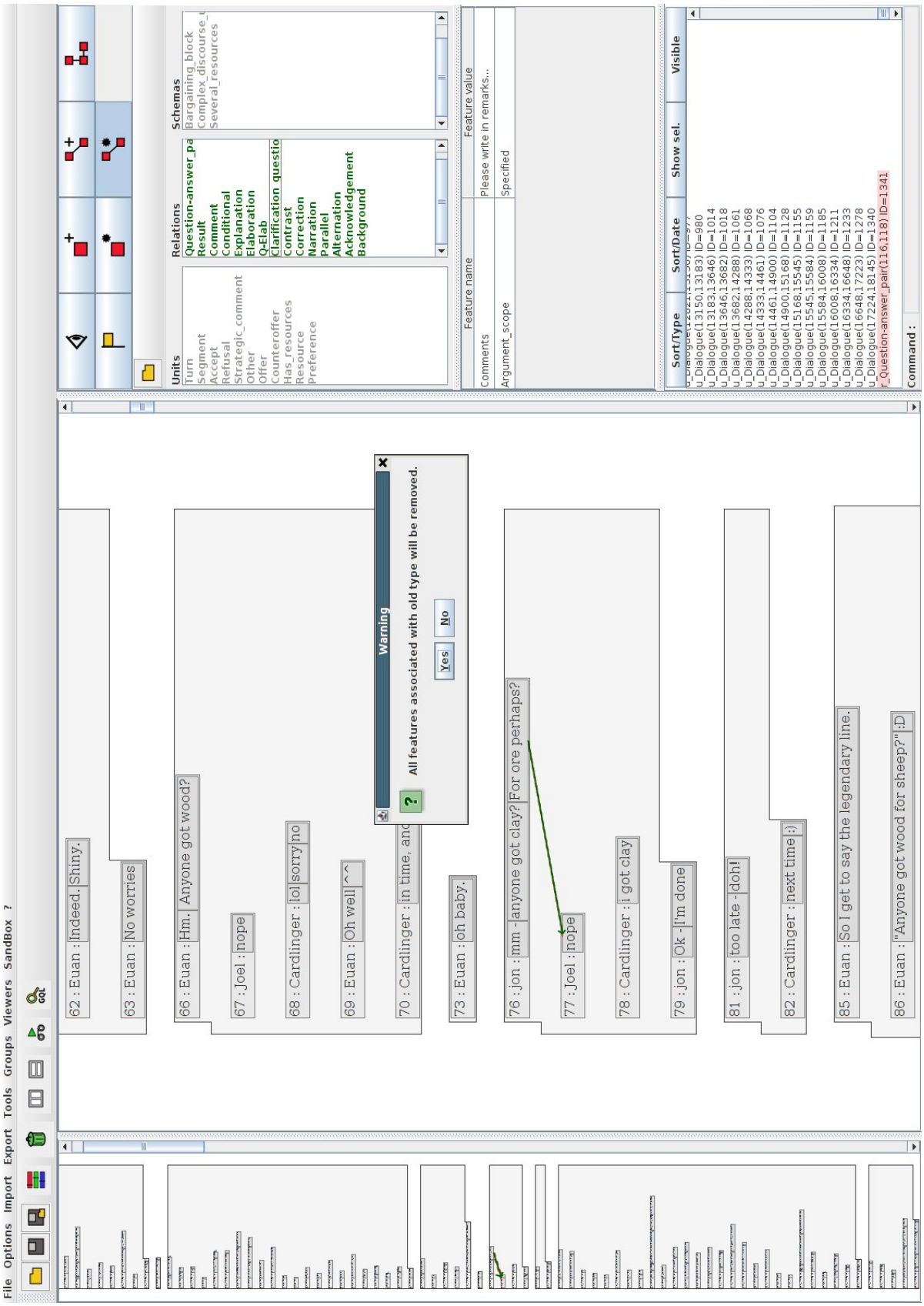

Figure 4: Editing a rhetorical relation annotation in Glozz

schemata are set as appropriate. Please pay attention to the fact that any "Several_resources" schema remains inside a (customized) "Segment" unit, viz. inside an "Offer" or a "Counteroffer" unit.

For creating and annotating all three types of schemata, you should follow the same general procedure, as described below:

1. in the Annotation editing toolbar (D.1 on Figure 1), click on the button 
2. click on the appropriate type of schema in the Item selection frame (D.2 on Figure 1), viz. "Complex_discourse_unit", "Bargaining_block", and "Several_resources".
3. create a new schema, by clicking on  , then on  , and then on all the units concerned (viz. "Segment" – in their customized form, "Turn" and "Resource"). For recursively aggregating more than two "Resource" units into a "Several_resources" schema, it is sometimes necessary to add a "Several_resources" schema to another such schema. In this case, please click on  before adding the schema.

In order to create other schemata, it suffices to repeat steps 2 and 3 above. Please also pay attention to the fact that "Complex_discourse_unit" schemata should group customized "Segment" units only; that "Bargaining_block" schemata should group "Turn" units only; and that "Several_resources" schemata should group "Resource" units only.

For editing a schema, similar steps as above should be followed:

1. edit the schema, by clicking on 
2. click on the red dot  , to the left of the main text window (for "Bargaining_block" schemata), to the right of the main window (for "Complex_discourse_unit" schemata) or inside an "Offer" or "Counteroffer" (for "Several_resources" schemata), which corresponds to the schema due to be edited
3. in the Editing Zone, click on the button 

4. (a) left-click on  , and then on all the units due to be removed; for removing a schema,  should be clicked on instead.

   (b) or, alternatively, on  and then on all the units to be added; for adding a schema,  should be clicked on instead.

Finally, any unit, relation or schema annotation can be deleted, by selecting it via the corresponding editing procedure as indicated above for each type of item, and then pressing on the `Delete` button of the keyboard (or `Suppr` for French keyboards). Concerning schema deletion, please note that the deletion of a schema annotation does not entail the deletion of its constituent units' annotations.

Saving the annotations

In order for the annotations to be easier to handle, conversations should be annotated in two steps:

1. first, annotations at the "Segment" and "Turn" level should be performed (that is, all annotations that do not pertain to discourse structure – i.e. CDUs and relations);
2. secondly, annotations that pertain to discourse structure – i.e. CDUs, and relations, should be performed.

In order to save one's annotations, click on the button  , then name the annotation file as follows:

1. for all annotations at the "Segment" and "Turn" level, performed in step 1 as shown above, the annotation file should be named:
   `FileName_AnnotatorName_u__DDMMYYYY.aa`, e.g.
   `pilot01_Smith_u__24042012.aa`;
   here, 'u' comes from 'unit';
2. for all annotations at the discourse level level, performed in step 2 as shown above, the annotation file should be named:
   `FileName_AnnotatorName_d__DDMMYYYY.aa`, e.g.

`pilot01_Smith_d_24042012.aa`;
here, 'd' comes from 'discourse'.

If one wants to just visualize the conversations and their annotated units, without taking the risk of altering the annotations, one can left-click on the button  and then navigate through the annotated conversations. Note however that in this way one is not able to see the feature structures associated to the units, relations and schemata. To do so, one must left-click on the editing buttons for each of these items.

Handling the annotation files

Given that the games contain a lot of turns, handling them at once can entail a slowdown of the Glozz annotation tool. If this is the case, you can annotate each game in smaller chunks, so that each chunk contains roughly one tenth of the entire game.

Thus, instead of loading one pair of (.ac and .aa) files for the whole game, several such pairs will be loaded and annotated, one after the other. Each such pair contains a number like $\_i$ in its name, just before the extension, e.g. `pilot01_2.aa` / `.ac` for the second chunk of the entire `pilot01` game. Thus, upon saving the segment-level annotations, the file name will be:
`pilot01_2_Smith_u_05062012.aa`. Upon saving the discourse-level annotations, the file name will be:
`pilot01_2_Smith_d_05062012.aa`. Each chunk contains at least one `Dialogue` unit, but can contain several such units. No `Dialogue` unit is split between two (or several) chunks.

Besides this detail, all the other conventions previously presented in this manual should be carefully observed.

5.3 Other Issues

As mentioned above, discourse structures can be represented as directed acyclic graphs. We don't assume that discourse structures are (mathematical) trees. The brief discussion below addresses those cases in which our discourse structures encode non-treelike data structures.

5.3.1 Two utterances related by multiple relations

In a discourse structure two EDUs may be connected by more than one rhetorical relation. The example in (75) illustrates this:

(75) a. Bush supports big business.
b. He's sure to veto House Bill 1711.

The EDU (75)b provides evidence for (75)a. In addition, a cause-effect relation holds between the eventualities introduced in each utterance. An interpreter might therefore assume that (75)b is a *Result* of (75)a, in addition to providing *Evidence* for it. This interpretation is encoded in our annotation scheme as follows:

`Evidence`((75)a, (75)b)
`Result`((75)a, (75)b)

5.3.2 A single utterance with multiple parents

A discourse unit may have multiple parents. The multi-speaker dialogue in (76) provides an example of this situation. B's utterance in (76) acknowledges A's offer in (76)a. A then asks a follow up question to B's acknowledgment that in fact also elaborates his original offer.

(76) a. A: I'm offering sheep for ore.
b. B: OK.
c. A: How about 1 ore for 2 sheep?

The graph for (76) is shown in Figure 5.

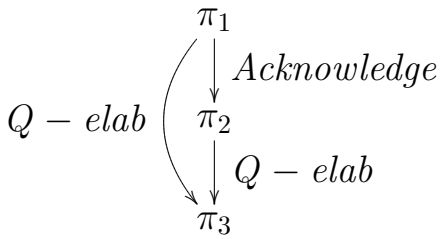

Figure 5: SDRS graph for (76).

There is no problem encoding such a structure in our annotation scheme:

6 Loose Ends/Conclusion

Things that need to be addressed or improved upon in the manual.