Chapter 8

Conversation Analysis and Specification

Anthony Finkelstein and Hugo Fuks

8.1 Introduction

We may crudely distinguish two broad approaches to the software development process, each of which views specification in a slightly different way.

The dominant research approach is that of formal development. In this approach a high-level abstract specification is made in a precise and analysable form, generally using discrete mathematics or logic. The description is verified and validated through the application of automated reasoning. An executable system is derived by the use of correctness-preserving transformations, automatically applied where possible. Subsequent system maintenance is done by changing a high-level specification and replaying the transformations to re-generate the system.

The approach on which most industrial practice is based follows a staged model of the software development process in which there are distinct requirements and design phases, each of which is completed when a specification is delivered. 'Formatted' representations, generally diagrams, with loosely defined syntax and semantics are used to support each phase. Diagram editing and consistency checking tools can be used to verify the requirements and the design. Validation is by inspection and walkthrough.

What both approaches have in common is the central rôle that they ascribe to specification. The construction of an exact description

Conversation Analysis and Specification
in Luff, P., Gilbert, N. & Frohlich, D.; Computers and Conversation, chapter 8, pp 183-187, Academic Press 1990

of software services, and the constraints under which those services are provided, is critical to the effective development of that software.

'Specification-in-the-large', that is the development of specifications for systems of substantial complexity and scale, mirrors 'programming-in-the-large' in raising a variety of difficulties that lie beyond the (non-trivial) clerical problems of handling large amounts of information. One such difficulty is that of specification from multiple 'viewpoints'.

'Specification-in-the-large' is, an activity in which there are many participants – clients, systems analysts, engineers, domain experts and so on. Each has differing perspectives on and knowledge about the object system, as well as a variety of skills, rôles and so on. In some cases these perspectives may be based on contradictory understandings. To construct a specification the participants must cooperate, that is, contribute to the achievement of a joint goal.

Existing specification schemes, methods and tools are generally based on specification by a single participant and refined using examples that consolidate this weakness. Our research objective is to develop a detailed (and, where possible, formal) understanding of specification by multiple participants so that we can both support the construction of specifications and reason about the process of specification itself.

We have constructed a preliminary model of specification by multiple participants in which the development of a specification is viewed as a conversation. The model deploys some formal apparatus, dialogue/commitment logics, taken from work on the foundations of logic, and an approach, cooperation and negotiation, from work on distributed artificial intelligence. We are extending this model using techniques and insights drawn from conversation analysis.

In this paper we will briefly discuss why we believe an examination of conversation may shed light on the process of specification. We will illustrate the development of a dialogue (or discourse) analysis based model and give a sketch of how it can be extended using conversation analysis. A short account is given of observational studies and specification support tools related to our approach.

The paper is based around an extended example. The example is not taken from a live software development project but has been constructed to illustrate our approach. This will not be our only violation of the spirit of conversation analysis.

The example reflects our interest in what might be termed 'fine-grain' software development modelling. By fine-grain software development modelling we mean the analysis and description of the detailed structure and organisation of development activities. In general this structure and organisation is ignored by those who are concerned with modelling software development at the level of tool invocation and inter-working (Kaiser, Feiler and Popovich 1988). We suggest that many important gross features of software development such as verification, validation and cooperation (Finkelstein 1989a) arise from the complex interplay of fine-grain activities. These features of software development are not simply embedded in a matrix of routine 'house-keeping' tasks. Rather, they are emergent properties that derive from the underlying fine-grain organisation.

8.2 Why conversation?

Before introducing the model it may be useful to examine the intuitions that prompted us to look at conversation in the first place.

The most significant informal motivation for using an understanding of conversation to develop a model of the construction of specifications is readily available. If we observe, naively, what happens during the process of specification we notice three features: participants talking to each other, participants writing documents, participants exchanging documents.

The setting for these activities generally consists of client(s) and specifier(s) sitting face to face across a table – the client explaining the requirements and the specifier occasionally asking guiding questions, seeking clarification, pointing out inconsistencies and raising unanticipated consequences. Interleaved with this spoken interaction are periods during which the requirements are documented. Throughout the process documents are exchanged, some containing further details of requirements on the part of the client, others containing preliminary documentation prepared by the specifier.

The spoken interaction, which appears to occupy the bulk of time devoted to specification, is readily recognisable as conversation. It may in addition be interesting to view the exchange of documents as a silent (but expressive) dialogue.

The specification process described above uses natural language. Hence, building representation schemes and selecting the appropriate conceptual categories for them, based on the way in which

requirements are expressed in natural language, appears to have been a good way to develop schemes with higher expressiveness (Balzer, Goldman and Wile 1978). The approach is exemplified by "Gist" and "Pure Tell" (Horai, Saeki and Enomoto 1987). It is not a great leap of the imagination to extrapolate from this to using the structure of conversation as an overall setting.

Conversations mirror specification in that they both display physical distribution and cooperation through the interaction of multiple physically and logically independent participants.

Conversation, like the specification process, is flexible and dynamic. The shape of a conversation is not typically determined in advance but evolves in the light of issues brought to the fore during the course of that conversation. Participants continuously make moves to realise their objectives based on the revealed state of play. Contributions to the conversation make 'sense' only in their immediate conversational context.

8.3 Example

Jeff, Tom, Mike and Sue are involved in the specification of software to support the preparation and assembly of user manuals for a range of widget production tools. Jeff is a technical editor and knows all about how documents are prepared. Tom is the librarian. He knows all about the categorisation, version control and storage of documents. Mike is a salesman. He knows all about how manuals are assembled from the collected documents to meet the requests of customers. Sue is a software engineer. She knows about the needs of the software designers and, because she is experienced in dealing with technical documentation systems, about some possible problems which may occur in such systems.

Jeff, Tom, Mike and Sue meet to write the specification. After introductions and preamble the meeting gets underway. We join it some time after its start. On occasion, we may need to present fragments of the conversation out of their temporal order so as to illustrate important features of our analyses.

8.4 Dialogue analysis

Our analysis of the example begins with a preliminary model based on dialogue analysis. The techniques we use are largely drawn from dialogue and commitment logics (for a more detailed discussion of these, see Hamblin 1971 and Fuks, Ryan and Sadler 1989). Similar approaches have been developed by Mackenzie (1981, 1985) whose work has formed a basis for the model.

```
[A] Jeff:   Documents can either cover new
            features or be a modification of a
            document that has already been
            issued.
```

In our preliminary model we distinguish two parts of the locution act [A]: a locution act modifier (in this example an assertion, it is the case that documents can either...), and a statement (in this example, documents can either cover new features or be a modification of a document that has already been issued). Our model uses the following modifiers:

```
assertions, to be read as "It is the case that
Statement";
denials, to be read as "I deny that it is the
case that Statement";
questions, to be read as "Is it the case that
Statement?"; withdrawals, to be read as " No
commitment to Statement";
challenges, to be read as "Why is it to be
supposed that Statement";
resolution demands, to be read as "Resolve
whether Statement" or "Resolve a potential
inconsistency in your commitments".
```

Statements are constructed in a propositional language which includes negation, conditional propositions and conjunctions of statements.

```
[B] Tom:    All documents are assigned a
            category.
```

[B] can also be broken down into a statement and a locution act modifier, but we can make some further observations. Tom follows Jeff and there has been a change in rôles. Jeff, who was the speaker, is

now the hearer (assuming he has not left the room) and Tom has assumed the rôle of speaker. We model this as a succession of locution events in which each locution event is represented by a triple:

<Stage [marks the progress of the dialogue, stage, stage+1 and so on], Name [of the current speaker], Locution Act>

[C] Sue: Are modified documents assigned a category?

[D] Tom: No, modified documents are not assigned a category.

In [C] Sue asks a question, is it the case that...?. It is immediately followed by an answer in the form of a denial. We note that there are syntactic relations between these locution events. It would not make any sense for Tom to follow Sue's question by a statement on some completely different topic, or in certain circumstances by another question. We model these syntactic relations in the form of dialogue rules. Thus in the example above, the following rule applies:

Dialogue rule Quest (Questions):
After the questioning of a statement (**questions**(Statement)), the next locution event must be either the confirmation of that statement, or the withdrawal or the denial of that statement (**asserts**(Statement), **withdraws**(Statement) or **denies**(Statement)).
No legal dialogue of length stage+1 contains a locution event
<stage-1,hearer,**questions**(Statement)> unless it also contains an event
<stage,speaker,**asserts**(Statement)> v
<stage,speaker,**denies**(Statement)> v
<stage,speaker,**withdraws**(Statement)>.

[E] Tom: There may be many current releases of any given document.

[F] Mike: A manual consists of all the current releases of documents in the categories associated with the tool owned by the client making the request.

[G] Sue: So, duplicate documents are allowed in a manual!

What is happening in this exchange? Tom makes a statement [E] which is heard by Mike, Sue and Jeff. Nobody objects and Sue feels it is safe to assume that everybody is committed to that statement and its consequences – duplicate current releases. In other words, Sue feels that Tom, Mike and Jeff have 'signed-off' the statement and taken some responsibility for its consequences (Winograd and Flores 1986; Thimbleby 1988). We note that locution acts, such as the assertion of a statement, establish (and remove) commitments. This is modelled in the form of commitment rules which define the relation between locution acts and commitments. A simple instance of such a rule might be:

```
Commitment rule W:

After a withdrawal the statement is removed from the
speaker's commitment store, the hearers store remains
unchanged.
After <stage,speaker,withdraws(Statement)>
      committed(stage+1,speaker)=
            committed(stage,speaker) - {Statement}
      committed(stage+1,hearer)=
            committed(stage,hearer)
```

Mike now makes a statement about how a manual is constructed [F]. Sue thinks she has spotted a potential inconsistency and tries to probe it by attempting to get Mike to commit himself to the statement that duplicate documents are allowed in a document [G]. The conversation continues.

```
[H] Mike:   No, I am not committed to duplicate
            documents being allowed in a
            manual.
[I] Sue:    Well in that case resolve your
            inconsistency with Tom.
[J] Mike:   I no longer hold that there may be
            many current releases of any given
            document.
```

In [H] Mike refuses to accept (effectively withdraws) the direct consequences of his commitments and Sue pounces by demanding that Mike resolves his inconsistency as in the following rule:

Dialogue rule Resolution (abbreviated):

The participant must adjust his commitments if s/he withdraws/challenges an immediate consequent of his/her commitments.

No legal dialogue of length stage+1 contains a locution event

<stage-1,hearer,**resolve**(ConjunctionOfStatements)>

unless it contains an event <stage,speaker,Act> and either:

locution Act is **withdraws**(Statement) and statement is one of the conjuncts of ConjunctionOfStatements.

or

locution Act is **withdraws**(Statement) and statement is one of the conjuncts of the antecedent of ConjunctionOfStatements.

or

locution Act is **asserts**(Statement) and statement is the consequent of ConjunctionOfStatements.

Mike resiles from his previous position [J]. This now leaves Mike free to challenge Tom and sort out the confusion over current releases.

[K] Sue: Why are modified documents not assigned a category?

[L] Tom: Because modified documents are automatically given the category of the originating document.

In [K] we can observe Sue challenging, a special way of requesting an explanation of, a statement about document modification. Tom provides an answer to the challenge [L]. Sue now knows (and is committed to) that modified documents are not assigned a category and that modified documents are automatically given the category of the originating document. She also knows that the two are logically related: the fact that modified documents are not assigned a category implies that modified documents are automatically given the category of the originating document. In our preliminary model, the way in which these logical relations are established and maintained is given in argument forms which are embedded in special commitment rules. For instance:

```
Commitment rule G:
After an assertion (AnotherStatement) which occurs as
a reply to a challenge (why(Statement)) both
participants are committed to the reply
(AnotherStatement) and to the conditional
(AnotherStatement -> Statement).
After <stage,speaker,asserts(AnotherStatement)>
where the preceding locution event was
<stage-1,speaker,why(Statement)>

  committed(stage!1,speaker) =
    committed(stage,speaker) U
    {AnotherStatement, AnotherStatement -> Statement}
  committed(stage+1,hearer) =
    committed(stage,hearer) U
    {AnotherStatement, AnotherStatement -> Statement}

      [M] Sue:    Why are modified documents
                  automatically given the category of
                  the originating document?
```

Our preliminary model is extremely sparse. By basing our approach on what is in essence a formal model of argumentation there are, aside from the well known difficulties of dialogue analysis, practical limitations in both the underlying language and the features of conversation we can capture. It is possible to build *ad-hoc* extensions. For example, in [M] we can see Sue making another challenge. This repeated request for an explanation, familiar to parents of small children, is a typical elicitation tactic. Tactics of this form could be modelled as special program-like scripts. For instance:

```
procedure refine;
begin
      repeat
            why;
            assert
      until
            goal
end;
```

Unfortunately, extensions of this form tend to introduce more problems than they solve, such as, in the case of this example, how are tactics identified, and what constructs are appropriate for such a scripting language.

8.5 Conversation analysis

It has been an interesting, and unexpected, result of our work on applying dialogue analysis to specification conversations that a very limited model can account for a large range of gross features of that activity. These include simple elaboration and verification tasks. The formal basis of our preliminary model has given us a systematic foundation for understanding specification.

What we seek from conversation analysis is not an alternative to the approach established above but rather a principled basis for its extension, in particular, to take account of conversational strategies and the more complex features of turn-taking in multi-party conversation. We believe the conversation analytic approach has a useful contribution to make in this area and seems to fit well with the 'philosophy' of what we have termed fine-grain software development modelling.

Contrary to the assertions of conversation analysts, for example Levinson (1983), we see no necessary contradiction between conversation analysis and dialogue analysis. It is worthwhile briefly examining the two major methodological points of departure between the approaches to clarify our position.

Conversation analysis is primarily an inductive enterprise which works directly from observational data, while dialogue analysis tends to impose a rigid external theoretical framework on those data.

It seems clear that software development, and specification specifically, is a particularly complex human activity (in this regard it can usefully be compared with the day-to-day talk that has usually been the subject of conversation analysis). Empirical studies must therefore attempt to balance carefully the dangers of coercing data to fit a model which is inappropriate, with the alternative of generating vast amounts of data (typically, verbal protocols) with no analytical framework within which to interpret it. Our judgement is that the pendulum in studies of software development has swung too far towards the latter. There is, however, a well developed canon of general principles and concepts derived from observation in the conversation analysis style which, we contend, may be safely applied to specification independently of direct investigation and observation.

Although it is true in principle that dialogue analysis theories are essentially unfalsifiable, there being no explicit procedure for

assigning a particular modifier to an observed locution act, the importance of this depends crucially on the primary objective of our work. Our objective is not explanation or prediction but rather developing automated support for specification conversations (this is very much in the same spirit as Frohlich and Luff and of Cawsey, in this volume). We are, in the long term, concerned with prescribing, constraining or guiding specifiers to act in accordance with suggested best practice in the conduct of the specification activity. The 'cost' of falsely or inappropriately identifying a sequence of events as ill-formed is simply the marginal cost of acting in a less than optimal way. Continuing the example:

```
[N] Sue:    So, all old document versions are
            stored because they may be needed
            in the future.
```

It is not clear that conversation analysts have successfully answered the argument that conversation analysis is based on an implicit categorisation of locution acts. However, the findings of conversation analysis point to a richer and highly task dependent and social situation dependent categorisation of such acts. In [N], rather than simply viewing what Sue says as an assertion, we could classify it as a summarisation with an associated commitment rule (possibly derived from that assertion). Other examples of such a categorisation might include criticism, posting an alternative, and so on.

```
[O] Tom:    Each modification of a document
            creates a new version of that
            document and the most recent
            version of the document is the
            current release of that category.
```

Our preliminary model assumes that each locution act has associated with it a statement which is a formula, which itself may consist of a conjunction of formulæ. In principle (note that some redefinition of the concept of immediate consequence may be required) each formula would constitute a turn constructional unit with an associated transition relevance point. This can be seen in [O]. Given this approach we can adopt turn-taking rules derived from conversation analysis (see Heritage 1987) more or less wholesale rather than the naive 'turn and turn about' rules in our preliminary model. At our current level of concern, complex features of turn-taking such as overlaps can, we feel, safely be ignored.

Conditional relevance follows naturally from any systematic study of conversation. In our preliminary model it is embedded in the

dialogue rules. The conventional conversation analysis adjacency pair organisation (question [C] – answer [D], and so on) will also be subsumed in the dialogue rules. Insertion sequences, of the type commonly identified in conversation analysis, are not explicitly recognised in our preliminary model. We include these by enriching our simple notion of locution event with the relative location of the locution act.

```
[P] Jeff:   I write all documentation.
```

Perhaps the most interesting feature of conversation analysis is the concept of preference organisation. In everyday conversation (inviting people to dinner, and so on) a preferred locution is one which builds, or tends to build, social solidarity while a dispreferred locution act destroys, or tends to destroy, social solidarity. Participants will delay dispreferred locutions. We might build an equivalent organisation in specification conversations. A dispreferred locution destroys, or tends to destroy, 'specification solidity' while a preferred locution promotes it. Instances of dispreferred locutions might be the introduction of exceptions, withdrawals and demands for the resolution of an inconsistency. Preferred locutions might be the "white lies" observed by Balzer (1985) in which a simplified statement is made to give the specifier a basic understanding and subsequently corrected to give a more accurate description. [P] is an example of this. We know that Jeff does not write all the documents, for he often modifies existing documents. Jeff cuts out this unnecessary complication by telling a white lie.

```
[Q] Jeff:   Well actually...
```

Specification as an activity is concerned with very precise communication and assurance of mutual understanding. In this context the suggested preference organisation is the skeleton on which sophisticated repair work can take place (see Raudaskoski in this volume). In the case of white lies the repair may be self-initiated [Q] while in verification-style repair it is other-initiated.

Where a dispreferred locution (or in special cases a non-normative locution) occurs, conversation analysis suggests that an account or explanation follows. This can be readily accommodated within the argument form framework above.

We have sought wherever possible to keep clear of the semantic analysis required to understand and use topic coherence (Hobbs 1982) in the control and organisation of conversation. By using a more expressive logical scheme as the underlying language in which

statements are built we may provide the hooks for a formal analysis of discourse structure.

8.6 Automated support

We aim, as software engineers, to provide automated support for developing specifications.

Such support has both a short-term and a long-term rôle to play in our work. Our short-term goal is to use it as a workbench while developing an improved understanding of the analytical tools and enhancing the descriptive schemes. Without support even relatively simple examples are awkward to handle. Our long-term goal is to use it as the core of a specification support environment (Finkelstein 1989b). In such an environment replaying a 'development history' would be equivalent to running through a record of conversation events (this can be compared to the approach of Conklin 1989).

We have developed two dialogue support systems (IC~DC One and IC~DC Two) which animate, albeit in a simple-minded way, the preliminary model. These tools allow the user to develop simple 'conversations' and then replay them in whole or in part. IC~DC One is written in Prolog and has been used to help us to understand and enhance the dialogue rules. IC~DC Two is written in Smalltalk-80 and has been used to investigate an appropriate architecture for a specification support environment. It is the vehicle for the proposed conversation analysis extensions to the model.

In both tools the dialogues are monitored for legality, and illegal dialogues can be explained and rolled back to a legal state. Users may view the commitments of the participants and may change the course of the dialogue by editing these commitments directly.

8.7 Observational studies

To date we have carried out no extended observational study of specification in its natural, that is industrial, setting, though we have carried out informal studies of university research teams developing small specifications of lift systems, central heating controllers and the like. Because of the commercial sensitivity of software specification, organising studies in natural settings is extremely difficult. Other

major methodological barriers are that: specification often takes place over a prolonged period; the most interesting exchanges take place some time after the specification phase is considered finished, that is, during implementation and post-installation maintenance; and conversations often hinge on documents, documentary references, back-of-envelope sketches and diagrams.

We have found it useful to refer to studies of specification practice in the literature, for example, that by Fickas, Collins and Olivier (1987). There is still some way for us to go using this data before the need to carry out our own studies becomes pressing.

8.8 Conclusion

We have seen in very crude outline how we might begin to model the 'fine-grain' of specification using a dialogue analysis model (for more detail, some fully worked examples and an account of the limitations and shortcomings of our preliminary model, see Finkelstein and Fuks 1989) and have given a sketch of the extensions that conversation analysis suggests.

As yet our work is at a preliminary stage. It should be stressed that there remains a substantial amount of foundation work to be done in order to make extended models of conversation that are both computationally tractable and formally sound. Future work includes the construction of a dialogue-based framework for theorem proving and a detailed analysis of the structure of elicitation and verification strategies.

Acknowledgements

The authors would like to thank their colleagues and students who have contributed significantly, through lively critical discussion, to the work this paper reports. Hugo Fuks is supported by the Brazilian National Research Council CNPq, grant 202471/86-cc. Anthony Finkelstein is supported by EC Esprit, DTI IED and SERC.