Third SAM Workshop, 2002

Jon Bell

Doc. ref SD/BCG/LAN/SDL/02 June 28, 2002

1 Introduction

Having spent some time attending the recent SDL workshop in Aberystwyth, SAM 2002, I felt it was worth writing a brief report. There was little of immediate interest from the SoftFMEA point of view, so I shall not describe the workshop in any detail. However, it does seem worth writing something on the discussion arising from Edel's presentation and there were one or two points I feel are worth noting down from Eckardt Holz's keynote address. There are sections on each of these, followed by a brief section noting any other points I feel are worth noting. Other papers are in the proceedings, of which I have a copy, thanks to Edel providing an extra one.

2 Discussion arising from our work

As part of the panel discussion, Edel gave a presentation based on a short position paper discussing why SoftFMEA has decided not to adopt SDL [2]. This gave rise to a more interesting discussion than I expected, which did raise to one or two interesting points.

Edel's presentation made the following points: -

- CANbus messages are broadcast, for which SDL has poor support.
- For FMEA we might need to model failure mode behaviours, which the complexity of SDL process diagrams might make laborious.
- The automotive sector is familiar with Harel's state charts (Statemate), the most likely alternative to SDL.

She then drew the conclusion that SDL lacked any features of sufficient importance to this work to motivate a change from state charts.

It was pointed out that Statemate was no better for modelling broadcast messages, to which I replied it was no worse. There does seem to be some acceptance in the SDL community that something needs to be done here, as was noted in the SDL Forum society chairman's closing remarks.

The need to model failure modes rather confused the workshop (perhaps Edel's introduction to FMEA was a bit brief!) and it was felt that modelling such behaviour was unnecessary. It might well be in specifying new systems, but we clearly need to model such component behaviours for FMEA. It was suggested that SDL's exception handling might be useful here. This might warrant further investigation, but I am not convinced that it will help us much.

The fact that Statemate is more familiar in the sector is pretty unarguable. Nobody suggested any advantages of SDL sufficient to change, so the conclusions of the report on which the presentation was based still hold.

This presentation was based on oldish work, and since then I have become doubtful about how well SDL defines the relationship between the structural and behavioural views of a system, see [1]. There is a little more on this in section 4.

3 On the keynote address

Eckardt Holz's keynote address had the title *Combination of Different Modelling Techniques for Software Engineering* which suggests at least some overlap with the project. Any overlap was slight, however, the theme being the need to combine models in different languages, as these are used at different times and for different purposes in the design life cycle. Therefore there was nothing on different model types in the MBR sense, nothing on combining structural, behavioural and functional models. He was more interested in mapping between models in, say, UML, SDL and VHDL, to combine information arising from different design activities (such as requirements modelling) and different parts of the system, such as hardware and software elements.

He pointed out the reasons for using models and that the models generated for one phase of a project are the starting point for the next phase, but that they use different languages. He suggests you need a description of the whole system (for context of more detailed subsystem models), and interface specifications to define the interactions between open sub-models, for different views of the system.

He suggested three approaches for combining models built using different languages :-

- Language integration - use a single (complex) language that includes all necessary elements of different languages you might use.
- Translation - Adapt a limited concept space, so you use concepts common to all languages, allowing translation between them.
- Co-operation - do the combination at a concept level, using a meta-

model or an abstract grammar. Avoids dependency on concrete notations.

He is interested in concept-space based combination. Models are made up of concepts and individual languages' notations represent all (or a subset) of these concepts. You might not have all concepts in each language, but you have common concepts as a foundation. Languages need to be defined in terms of the concepts they can be used to represent.

It is not clear that there was anything of direct value to the project's interest in combining models, it might be argued that we are interested in combining models expressly to allow different concepts to be captured - isn't this also the point of using different languages in software engineering?

4 Other items

The papers had little of any of interest to the project, but the competition was (mildly!) interesting. This was to model a railway crossing in SDL. Arguably, the requirements ruined it as an exercise in modelling, as they were so implausible. For example, the trains were to be modelled as part of the system, while road traffic was not. As the intended communication and sensing between the crossing and the trains and cars is similar (red and green lights and sensors) this requirement seems anomalous, and indeed the winner of the contest pointed out that this requirement was unrealistic. While this problem would be avoided if the setters of the contest knew how level crossings work (!), I an also tempted to argue that a language or tool with a well defined relationship between the structural and behavioural sides of the system being modelled would help avoid these errors. One contestant had trains telling the following train where it was and how fast it was going, without explaining how, that is by what medium. At least Statemate (for example) would expect some physical medium for transmission of these messages. All this left me with the impression that the SDL people are not unduly concerned about the correct relationship between the structural and behavioural sides of a system. I imagine this impression is rather unfair.

5 Conclusion

As will be seen from the notes on the discussion of Edel's presentation, nobody came up with any features of SDL which suggest we are wrong to prefer using state charts. In his summing up, Rick Reed noted that the SDL forum needs to tackle the broadcast question, one of the problems identified in Edel's presentation. It is perhaps not surprising that this point was noted, as problems with broadcast messages could occur rather closer to the SDL community's application domain.

My current feeling regarding adding broadcast messages to SDL is that it might need a more formal relationship between the structural and behavioural views of the subject system than currently seems to be the case. This difficulty seems to be illustrated by the unrealistic modelling of the railway crossing in which each train told its following train where it is and how fast it is travelling. It was never made clear how these messages should be transmitted. It could be argued that this apparent laxity would be discouraged by the presence of the structural view in Statemate. I have had one or two rather trivial idea here, which could perhaps be made into a small paper for the SDL people.

References

[1] Jon Bell. Modelling behaviour. SoftFMEA document ref. SD/TR/MM/01, 2002.

[2] Jon Bell, Edel Sherratt, and Neal Snooke. Sdl for automotive modelling: the challenge. Unsubmitted position paper for SAM '02, 2002.