The role of functional decomposition

Jon Bell

Doc. ref. SD/TR/FR/10: July 27, 2004

Abstract

Hierarchical decomposition of function is already a feature of the language used for interpretation of simulation for FMEA ad SCA. This report attempts to arrive at a more formal approach to its use, illustrated by examples. It attempts to tie in the idea of hierarchical decomposition with the idea that function be treated as "the intended purpose of a device, expressed in terms of properties of the device".

1 Introduction

This report considers the idea of hierarchical functions, suggesting a more strictly defined approach to a hierarchy of function than is used at present in AutoSteve and attempting an argument as to the rôle of a hierarchy of system function for interpretation of simulation in design verification and failure analysis. It also attempts some clarification of the nature of a possible boundary between behaviour and function.

The functional language used for interpretation of simulation results in AutoSteve uses the idea of a hierarchy of functions, composed of sub-functions. This decomposition of system function is (or should be) carried on down to the level of individual input or output components. This is related to the hierarchy of function in (Snooke and Price 1998) but that paper takes the idea further than was implemented, by considering component level functions that might or might not be replaceable in terms of the functioning of the system, such as the conducting function of a switch being replaced by it's being shorted out.

It should be noted that a car exterior lamps system has been used as an example for illustrating possible relationships between subsidiary functions. The legal implications have been ignored, throughout, however.

2 A definition of function

One difficulty with the four classes of knowledge listed by (Chittaro and Kumar 1998), and used in my "grid" diagram is the tendency to somewhat blurred distinctions between behaviour and function and function and purpose. This is exacerbated by different researchers using different (though not dissimilar) definitions of function. In this section I shall briefly discuss these definitions and attempt a possible set of definitions.

There are, according to (Chittaro and Kumar 1998) two alternative definitions of function. These are the operational definition, where function is a relation between input and output and the purposive, where function is a relation between user's goal and the component's (or system's) behaviour.

I propose the following as an informal definition of function.

The function of a device is its purpose expressed in terms of properties of the device.

By this is meant that a description of the function should identify the states of the device when it is fulfilling its intended purpose. It seems worth trying to fit this definition with appropriate definitions for the other classes of knowledge in (Chittaro and Kumar 1998). I propose the following:-

Structure The physical composition of the device; its components and connections.

Behaviour How a device works, what it does in terms of its internal properties.

Function A device's purpose expressed in terms of the properties of the device.

Purpose The need that the device is intended to fulfil.

With the suggested split between simulation and interpretation, there is a definite distinction between behaviour and function, behaviour being on the simulation side and function on the interpretation side. This might be affected if a functional model resulting from simulation of some system was used in simulation of a larger system of which the earlier system is a part, but even in this case, it is preferable, at least for failure analysis, to use a behavioural rather than functional model of the subsystem.

One possibly interesting approach to distinguishing between the four classes of knowledge is:-

Structure What is inside the device.

Behaviour What happens inside the device.

Function What happens at the surface of the device (or the boundary of the simulated world).

Purpose What goal is enabled outside the device.

This does draw a reasonably clear distinction between behaviour, function and purpose which seems not inconsistent with other approaches, though perhaps the idea of function as effect in (Chandrasekaran and Josephson 1996) is veering towards the outside viewpoint. I do not suggest these are complete as definitions, there is no idea of intention in these, though it is implicit in the idea of purpose. These viewpoints moving out from the inside to the surface to the outside of a device are certainly consistent with the definitions above. It is also consistent with the idea that each class of knowledge abstracts the previous. I remain uneasy with this notion because for failure analysis, at least, we need to consider alternative behaviours, which even if we map to unintended functions can hardly be sensibly mapped to unintended purposes. I suggest that we cannot simply replace the alternative behaviours with the idea that the function (or purpose) has failed, as we will want to know what did happen as well as what didn't but should have. There is more on the idea of component level function later.

3 Hierarchical functions

Following the proposed definition of function above, I suggest that the aim of a functional hierarchy is to map between system behaviour and purpose. Note that this means that several system outputs might be expected to occur together but not be a part of the same functional hierarchy. For example we might list some of the "top level" functions for a car lighting system as in Table 1. It will be seen that the inputs (preconditions if you prefer) make clear that whenever

purpose	output	input
car visible	sidelights lit and tail lamps lit	light switch position side or head
light road	headlamps full beam	light switch position head and dip switch position main
light road no dazzle	headlamps dipped	light switch position head and dip switch position dip

Table 1: Functions of a car lighting system

the headlamps are lit, so should the side lights and tail lamps be, although they do not really contribute to the same purpose - how can tail lamps possibly help light the road ahead!? It is appreciated that the headlamps do contribute to the "car visible" function. Arguably this point simply illustrates the idea that there will inevitably be a degree of subjective choice in deciding on a system's functional description, as will be further illustrated in this discussion. It might be argued that there is no real need to use a hierarchy of function here; it is sufficient, for example, to give the required output for the headlamps dipped function and left headlamp dipped filament lit and right headlamp dipped filament lit. This leads to the difficulty I have noted in previous reports, that as there is no function associated with a single headlamp being lit, its being lit unexpectedly is missed in interpretation of the simulation. There is, however, another advantage in refining the functional model using the hierarchical operators that I have not seen fully discussed, though it relates to (Snooke and Price 1998). This is the possibility that the design analysis tool can refine its assessment of the severity of some component failure by its effect on the hierarchy of function. A simple case is that if the function "light road no dazzle" is composed of the two sub-functions left dipped ad right dipped then the tool can report "function light road no dazzle failed because left dipped failed". This is very simple, and is already done, of course. It can also be argued that the failure of this function is less severe if it is caused by the failure of one but not both of the sub-functions - the driver could limp home if only one of the headlamps dipped properly more readily than if both failed (though I appreciate the legal aspects of this example). While this may not have been properly implemented, it is hardly new, but another aspect of functional hierarchy, which again leads on from (Snooke and Price 1998) is the possibility that alternative sub-functions can (at least partly) affect the severity of a top level function. Suppose we add a "light road" function to the top of our hierarchy of function for the lighting system, with the post-condition "light road main OR light road no dazzle". This allows the system to recognise that a failure resulting in loss of both headlamps' main beams is less severe than a failure resulting in loss of all headlamp output, as the driver can limp on by leaving the headlamps dipped. One could imagine cases where different values of severity are used for these different cases of failure of the same function.

It is appreciated that this refinement of the functional model does add a modicum of complexity, but I note that such refinements need not be used. Indeed, they are arguably inappropriate in this case because of the legal implication. Clearly the interpretation must be capable of tracing down the hierarchy of function to find the right level that a failure affects. This raises the idea that there are two interpretations of AND - cases where both subsidiary functions (or post-conditions) must be present, such as the headlamps case, and cases where failure of one of the subsidiary functions can be regarded as less severe than failure of both. A failure in the

belt minder system such that only the chimer fails but the warning lamp still comes on might, for example, be regarded as less severe than a failure that causes both to fail, so no warning is given. In the latter case, the interpreter will need to explore the function hierarchy to find out how the top level function failed, while in the former case this is unnecessary. Equally, in the former case, any failure of the top level function can be given the severity value associated with that function, while in the latter that value for severity might only be used if both the subsidiary functions have failed. If only one has failed, then its severity value might be used. Does this mean AND is overloaded? Two approaches to distinguishing between these cases occur to me. One is to only have a subsidiary function if it is intended that it alone is a useful partial fulfilment of the top level function. There are problems here, the first is the need to carry on the hierarchy until a single output is sufficient, so that the sub-functions are recognised as being achieved unexpectedly and the other is that there might be cases where a function must be described hierarchically. My approach to timing, where a flashing continues for a fixed time (for example) is a case in point. Another possibility is to only associate severity values to subfunctions (when they fail) when they are regarded as partial fulfilment of a higher level function. The problem here, of course, is that it might not be appropriate to allocate severity values at all. The alternative is some alternative operator, a sort of "loose AND" to distinguish cases where partial fulfilment of the top level function is better than nothing.

Another aspect of the hierarchical relations which results from the triple representation of function is how much more useful OR, XOR and NOT are in describing preconditions (inputs) than outputs. It is pretty hard to come up with a case study function whose post-conditions can e combined using OR, frequently they are really different functions. However there is no difficulty in doing so for preconditions. For example, the car visible function (that is fulfilled when the side lights and tail lamps are all lit) has as its precondition "lamp switch position side or lamp switch position head".

A function might be composed of alternative functions (though this is getting a bit messy!), or subsidiary functions. Which is required seems to relate to the logical relation concerned, though probably not exclusively. For example the required post-conditions (outputs or goal behaviours) for a function will typically be combined using AND (or the temporal operators) while alternative functions that achieve a purpose will be combined using OR. As noted above, it is pretty hard to invent a function whose outputs are combined using OR. For example, the "road head lit" function of a car lighting system can be fulfilled using headlamps dipped or headlamps main, but these are both complete functions, each with distinct variations on the (common) purpose and each with distinct pre- and post-conditions.

4 Component functions

One possible distinction that can be drawn between the ideas of behaviour and function is that behaviour is internal to the device (component or system) while function is an external view, treating the device as a black box. This leads to the idea that representation of function is only useful to the outermost layer of the system hierarchy. This in turn leads one to consider the question of how useful or interesting is representation of component function. There does not seem to be a simple answer to this question, it depends on the design analysis task concerned. If one is using functional decomposition and refinement to develop the design, as suggested by (Iwasaki, Fikes, Vescovi, and Chandrasekaran 1993), then the design analysis will clearly consist of comparing the qualitative (structure and behaviour model of the system against the functional decomposition. One obvious example is that if a wire's function is to conduct electricity of the left hand headlamp but it has been connected to the side light by mistake, this comparison

might show the error. I remain unconvinced that that model of design is not too simple - surely most designers will also use case knowledge to guide refinement of design.

For failure analysis, component function seems less useful. It seems possible to argue that knowledge of a component's function adds nothing to knowledge of its behaviour, knowledge of its place in the system's structure, and knowledge of how the system works. FMEA (and diagnosis) depends on knowledge of other possible component behaviours (such a switch sticking) and changes to structure (such as a wire shorting to ground). Functional knowledge does not seem to add anything to these, and indeed could be argued to hide such knowledge. If a wire's function is to conduct, then if it is shorted to ground then it is fulfilling its function. If its function is defined as conducting to some component then it fails when shorted to ground, but its faulty behaviour needs explicit representation, and treating this as function merely seems to add work, as the required knowledge to simulate this fault is contained in the change to system structure and the wire's known behaviour. A question that this leads to is how interesting is explicit knowledge of mode of deployment. Surely if the simulator has knowledge of the different behaviours of a component, it will automatically use the right one, in response to the operation of the system. To use the wire example from (Chandrasekaran and Josephson 1996) a wire can transmit a pull, conduct a current, generate a sound (if plucked) and so on. Is there any need for an explicit representation of which of these behaviours is to be used? If the input to the system is electrical the wire will conduct, if it gets pulled on it will transmit the pull etc. One other possibility is that a failure of the system might involve some other mode of deployment being used. A possible example is the (possibly apocryphal) stories of failure in an electric guitar giving an electric shock to the axeman through the strings through another of a wire's behaviours, conducting rather than sound generation.

I guess the difference is that the explicit representation of component function and mode of deployment are intended to allow use of functional knowledge in simulation. This might be necessary to overcome the limitations of the simulator (can we model the above mentioned failure of an axe using AutoSteve!? Actually we probably could with some bodging).

5 Behaviour and function

One area which maybe still needs some thought is how behaviour is mapped to function, how significant behaviours are identified. AutoSteve uses "output properties" for this. Have these been formally written up and if so, where? They sometimes seem to amount to labelling of a goal state (such as a remote locking system being in the state "locked") and sometimes to a way of highlighting some aspect of behaviour that is beyond the scope of the simulation, such as using the idea of a lamp being lit whenever there is current in the filament.

There is some similarity between this second sense of an output property and the "abstract states" that (Keuneke and Allemang 1988) see a need for. In that case, an abstract state is used to describe a system state dependent (principally) on some complex underlying behaviour, such as a cycle. In both this case and the second sense of output property, it could be argued that all that is being done is to side step limitations of the simulator. For example, an "ideal" simulator, with knowledge of temperature, properties of filaments, etc could tell that a lamp will glow when current flows through, so that the goal state (lamp filament being active) needs no further explanation. Similarly in the case of the electric buzzer used as an example in (Keuneke and Allemang 1988) if the simulator had sufficient knowledge of the behaviour of springs and the effects of electromagnetism, and causes of acoustic vibration it could establish that the "reed" will repeatedly close the circuit, and not only do so, but make a noise.

Limitations of simulators' knowledge do seem to suggest a need for such explicit labelling of

significant states, either because the effect is in some other domain from the simulator's and / or because of the need to abstract complex underlying behaviour.

Another (possibly rather academic) question that surrounds this is where function (in the sense of purpose expressed in terms of system properties) is separated from behaviour. This seems to depend on an arbitrary choice of the boundaries of the system. If the output or goal state is sufficient to describe the system's purpose (the purpose of a buzzer viewed in isolation is to make a buzzing noise when the button is pressed) then it seems reasonable to think of such a goal state in terms of function, while this becomes more difficult once the behaviour is that of a component within a larger system, so that in the case of the belt minder system, for example, buzzing is merely a requisite post condition for correct achievement of the warning function. This leads on to the idea that a system might be analysed in isolation and the relationships between pre and post conditions be used to model that system as a component in a larger system. Hardly new, I know!

All this does seem to illustrate the idea that the split between function and behaviour is hard to clearly define. Indeed, the split between structure and behaviour (or function where behaviour is not represented) is much the clearest of the divisions between the four classes of knowledge listed in (Chittaro and Kumar 1998) and used in my "grid" diagrams. However, I am reasonably persuaded that there remains a rôle for each of the three other classes of knowledge, as discussed in the section on defining function, Section 2.

6 Discussion and conclusion

References

Chandrasekaran, B. and J. R. Josephson (1996). Representing function as effect: Assigning functions to objects in context and out. In *Proceedings of American Association for Artificial Intelligence*.

Chittaro, L. and A. N. Kumar (1998). Reasoning about function and its applications to engineering. *Artificial Intelligence in Engineering 12*(4), 331.

Iwasaki, Y., R. Fikes, M. Vescovi, and B. Chandrasekaran (1993). How things are intended to work: Capturing functional knowledge in device design. In *Proceedings of 13th International Joint Conference on Artificial Intelligence*, pp. 1516–1522.

Keuneke, A. M. and D. Allemang (1988). Understanding devices: Representing dynamic states. Technical Report 88-AK-DYNASTATES, Ohio State University. From the Laboratory for Artificial Intelligence Research, Department of Computer Science.

Snooke, N. A. and C. J. Price (1998). Hierarchical functional reasoning. *Knowledge-Based Systems 11*(5–6), 301–309.