A Model-Based Approach for Organizing Quantitative Computations

J. Sticklen

A. Kamel

William E. Bond
Missouri University of Science and Technology, bondw@mst.edu

A Model-Based Approach for Organizing Quantitative Computations

Jon Sticklen and Ahmed Kamel
AI/KBS Laboratory - CPS Dept
A 714 Wells Hall
Michigan State University
East Lansing, MI, USA 48824
sticklen@pleiades.cps.msu.edu

W.E. Bond
Artificial Intelligence Group
McDonnell Douglas Research Lab
St. Louis, MO, USA 63166
bond@mdc.com

ABSTRACT

Model based reasoning (MBR) is currently receiving wide spread attention because it offers a way to circumvent the brittleness of reasoning systems built solely on associational knowledge. Initially MBR was explored under a general viewpoint of the envisonment process, although more recently, the field has broadened substantially. To date, most MBR approaches have focused on the use and manipulation of qualitative models. We report our experience in applying techniques of Functional Reasoning to the general problem of organizing quantitative calculations. As a testbed, we have solved a problem initially posed at the Model-Based Diagnosis workshop held in Paris, in July, 1989: representing an automotive cruise control system. Our results show that the principles of the Functional Reasoning Approach can provide leverage in device domains characterized by quantitative data. We end with a discussion of the current state of research in Model Based Reasoning.

1. INTRODUCTION

Model based reasoning (MBR) is recognized as offering a way to circumvent the brittleness of reasoning systems built solely on associational type knowledge. MBR is also attractive because it captures an intuition that is especially cogent in engineering areas: in order to troubleshoot a device, or redesign a device to new specifications, or ... a device, it is very useful to know how the device "works" — i.e., to have a model of the device.

The MBR field is quite heterogeneous, even on the issue of the basic building blocks for models. Models have been described in terms of structure and correct behavior of the device [1; 2; 3], behavior only [4], and those that represent structure and malfunction mode behavior [5]. In a highly related area, a large body of research has been undertaken under the heading *naive physics* [6]. There are two characteristics which generally describe previous research. First, the underlying goal has been to develop a qualitative mathematics in which one can minimally represent a physical device or situation, and which can be used to drive consequence finding over the device or situation. (This is well described in [7] for the case of the naive physics research.) Second, although there have been notable exceptions [8; 9], prior research has employed a qualitative approach to the modeling problem.

An alternative MBR methodology is the Functional Modeling (FM) approach. The goal of FM is to make use of known functionality of a device, to use that knowledge to *organize* causal understanding of the device, and to provide a reasoning algorithm which can be used to simulate the device for given starting conditions. The roots of FM lie in research by Sembugamoorthy and Chandrasekaran which set the initial representational concepts for the functional point of view [10]. Sticklen and Chandrasekaran applied and extended the initial work to include a simulation component to support diagnostic problem solving in a medical domain [11; 12; 13]. Goel has used a simulation viewpoint to attack problems of design problem solving [14]; Punch has likewise used a FM simulation point of view as a basis for the integration of Generic Tasks [15]. Allemang has recently reported an application of the methodology of functional representation to model the computer programs [16]. Finally, Keuneke recently completed a research project in which she demonstrated that the functional representation is a valuable framework for

An extended treatment of this material has been submitted to **Engineering Applications of Artificial Intelligence.**

the extraction of explanations of diagnostic conclusions [17]. Overall, the functional viewpoint centers on enumerating the proper primitives which can be used to organize causal device understanding. Similarly motivated research has recently been reported by Chittaro et al in Italy [18], and by Franke at UT-Austin [19].

To date, FM has not embraced a quantitative component. The research described here extends our previous work by integrating into FM the ability to perform numerical computations. Our work was motivated by an example suggested by Chris Price at the Model Based Diagnosis Workshop at IBM Paris, in July, 1989: the automatic cruise control (ACC) system of an automobile. The testbed example was stated by Dr. Price in generic terms; i.e., not implying any particular modeling technique.[1] After initial inspection of the ACC, we believed we could apply the then-current FM techniques to the problem. On trying to work out the details of the example however, we concluded that to deal with the problem in any sort of a "natural" manner, a quantitative capability was required. We then extended FM to include primitives for quantitative calculations. Most importantly, the extension led us to a new understanding of the functional representation and simulation: it is not intrinsically qualitative; given an example such as the ACC, it can be used as a framework for *organizing* necessary numerical calculations. We believe this is a new insight into the power not only of FM, but of Model Based Reasoning in general.

The rudiments on the Functional Approach have been described fully elsewhere (see [13] for a cogent tutorial), hence we will not reiterate that material here. Below, we describe our testbed (the automatic cruise control system), the extensions made in the FM approach, and a description of the functional model for the cruise control. We conclude with observations about our results and the general importance of good testbed problems such as the cruise control system.

1. Copies of the original problem statement can be obtained by request to Dr. Chris Price, Department of Computer Science, University College of Wales, Aberystwyth, Dyfed, SY23 3BZ, United Kingdom..

2. AUTOMATIC CRUISE CONTROL

The automatic cruise control (ACC) system is a hybrid system that automatically controls the cruising speed of the vehicle. It consists of electrical, electromagnetic, pneumatic, and mechanical components. At the top level, the ACC can be conceptualized as an equilibrium seeking system which seeks to eliminate the difference between two control signals: the command-speed signal set by the driver of the vehicle, and a signal indicating the vehicle's true speed. The organization of the ACC is indicated in Figure 1.

The ControlElectronics subsystem takes as input the two electrical signals (the command speed signal, and the feedback speed signal) and produces a control signal for the engine throttle actuator. The signal produced depends not only on the instantaneous value of the difference between the two input signals, but also on the history of this difference.

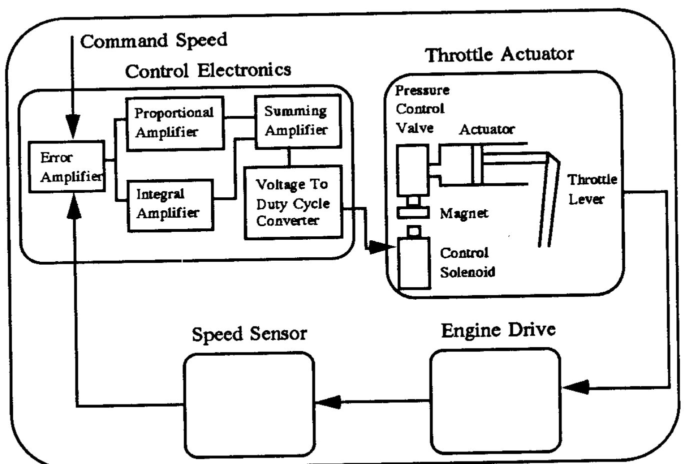

Figure 1: Cruise Control System

The ThrotleAccuator subsystem takes as input the throttle control signal and sets the engine throttle position. The transformation from an electrical signal to a mechanical one is done by an electromagnet controlling a mechanical valve. The movement of the pressure control valve affects the pressure inside the valve's chamber because of the arrangement of two outlets: one to the outside air and the other to the intake manifold (the vehicle's vacuum system). The pressure control valve is in turn connected to an air cylinder; the actuator cylinder. The pressure in the valve exerts a force on the piston of the actuator cylinder. This piston is also connected to a mechanical spring. The position of the piston is determined by the balance between the spring force and the force exerted by the air pressure inside the chamber. The throttle lever is directly coupled to the actuator piston, and thus the position of the actuator piston directly determines the position of the throttle lever.

The EngineDrive subsystem is the engine of the vehicle. We are concerned only with a small part of this subsystem: the setting of the vehicle's speed based on the engine throttle position. For the purposes of our example, the EngineDrive subsystem is treated as a black box, as is the SpeedSensor subsystem. We only deal with the high level function of the Speed Sensor system which generates an electrical signal having a voltage proportional to the vehicle's speed. This voltage is fedback to the control electronics subsystem.

The automatic cruise control system encompasses two types of representational challenges. First, the representation of the ACC should smoothly integrate low level understanding of the physical processes involved which are usually expressed in a quantitative manner (eg; a summing amplifier takes two inputs A and B and produces output A+B) with high level understanding of the purposes/goals of each of the major device subsystems (as expressed above). Second, the representation should provide some way to organize the numerical calculations which are necessary for performing a simulation of the ACC. This second representational challenge is actually an issue of control; i.e. of finding the proper calculation to carry out in a smooth manner. As we will argue later, the extended FM approach we have developed provides a framework such that both of these challenges are met.

3. Extending FM Approach

The FM approach consists of two sublanguages for description: one sublanguage for description of function, and one sublanguage for description of behavior (i.e.; a language of state variable change). To solve the ACC representational problem, we extended both sublanguages. (Keuneke has also suggested extensions for the function sublanguage [17] .)

The function sublanguage is very simple, consisting of only three parts: a precondition, a postcondition, and a pointer to implementing behaviors. Previously, we have represented the postcondition in terms of the primitive *ToMake*; i.e., the action of this primitive is to modify the value of a state variable of the device. In the ACC example, we need another primitive, one which will allow us to clearly indicate that the action of a function is going to be to fix a state variable *based on the context of other state variable values* at the point at which the function is invoked. We have called this new primitive of the function sublanguage *ToCalculate*.

Corresponding to the ToCalculate primitive in the function sublanguage, we also required a similar new concept for the state sublanguage. The representation of a behavior in the FM approach consists of a graph structure in which the nodes (after the first level nodes) represent statements about changes of device state variables. Until our experience with the ACC, these statements about state variable changes were of two types: setting state variables to some stated value, and incrementing state variables by some set amount.

To naturally represent the ACC we augmented our sublanguage for state by allowing "parametrized state change" in which a node in a behavior can be stated as a numerical calculation over other variables of the device which then sets a stated variable according to the result of the computation.

We illustrate our extensions to FM below be displaying parts of our representation for the cruise control system.

4. Functional Model of the ACC

4.1. FM Representation : ACC

The device decomposition of the ACC is shown in Figure 2. The device decomposition is a direct map from the physical structure of the ACC physical structure with one exception. We have split the physical control electronics subsystem into two subsystems: the control electronics subsystem and the error amplifier subsystem. This splitting was necessary to represent the system as an equilibrium seeking system at the high level.

Figure 2 shows the component decomposition of the ACC. The second step in applying the FM approach (following component decomposition) is to enumerate the functions of each device of the system, and the behaviors which implement these functions. To accomplish this, a three level graph such as shown in Figure 3 is developed. For example, at the highest level, the Cruise-Control-Sys-

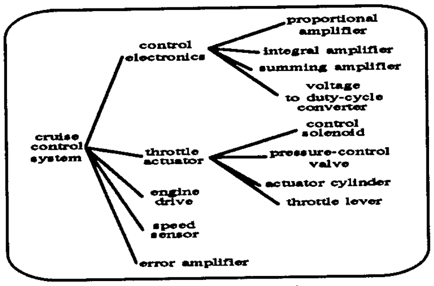

Figure 2: ACC Device Decomposition

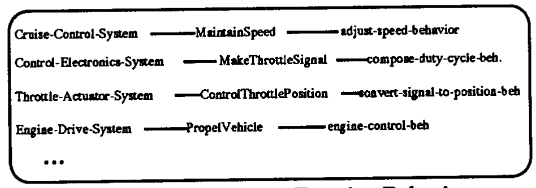

Figure 3: ACC Device/Function/Behavior

tem level, there is one functions: the *MaintainSpeed* function. The *MaintainSpeed* function is implemented by a single behavior, the adjust-speed-behavior.

The first step of the FM methodology is device decomposition. To represent the ACC, no extensions to our previous work were need to accomplish this step. The second and third steps of the methodology are to represent the abstractly stated functionality that is known for each device, which amount to listing its preconditions, postconditions, and listing a pointer to its implementing behavior(s). The third step of the methodology is to represent each behavior as a state change graph such as shown in Figure 4. It is to accomplish this third step that we introduced the *parametrized state variable*, as described below.

Below we describe the representation developed for the ACC. As is true for any FM representation, the ACC representation should be viewed as a set of causal chain fragments which, taken together, give an organized view of causality associated with a physical device.

Let us start with the highest level behavior of the cruise control system, the adjust-speed behavior. As shown in Figure 4, if the speed of the car does not match the speed the driver has set (i.e., if the error signal is not zero), then a number of causal consequences will follow in a set sequence. Note that by looking at one graphic (Figure 4), it is possible to grasp the overall operation of the cruise control system.

The first causal consequence in Figure 4 is that a new value of the "duty cycle" is calculated. The reason that causal consequence occurs can be ascertained by following the link *MakeThrottleControlSignal* function of the control electronics subsystem, which is shown in Figure 5. Similarly, by a process of following the annotated links, it can be seen that the make-duty-cycle behavior which implements the *MakeThrottleControlSignal* function (shown in Figure 5). Going to a yet deeper level of detail, Figure 6 shows the implementing behavior make-duty-cycle-behavior.

Two more layers are shown in Figure 7 and Figure 8. There are several key points that should be emphasized about the representation of the ACC that we have shown in part in Figure 2 through Figure 8. First, a FM representation is modular. Causality is represented in small chunks that chain together via annotations of why one causal chain follows another. Second, a FM representation "bottoms out" at a point that is appropriate for the problems the model must address. If our model of a cruise control were to be used to trouble shoot sub-chip level devices,

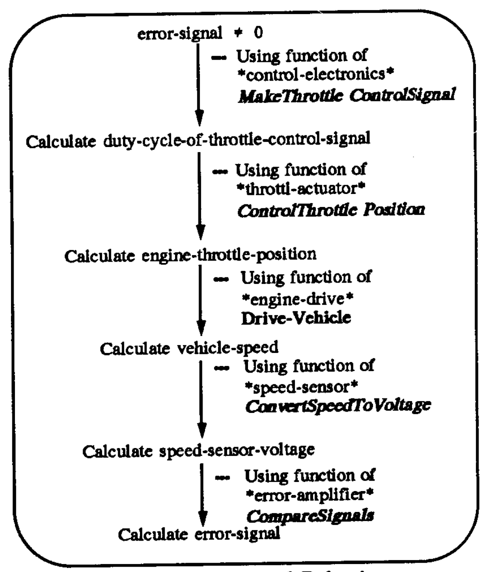

Figure 4: Top Level Behavior-adjust-speed-behavior

ToCalculate:
duty-cycle-of-throttle-control-signal

Provided: (error-signal ≠ 0)

By: make-duty-cycle-behavior

Figure 5: *MakeThrottleControlSignal*

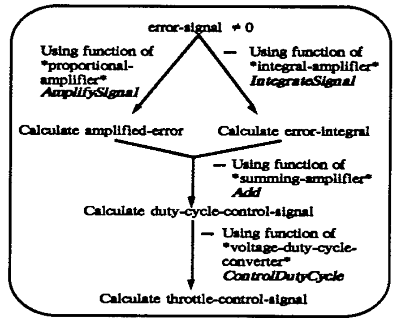

Figure 6: make-duty-cycle-behavior

ToCalculate: amplified-error

Provided: (error-signal ≠ 0)

By: amplification-behavior

Figure 7: AmplifySignal of proportional-amplifier

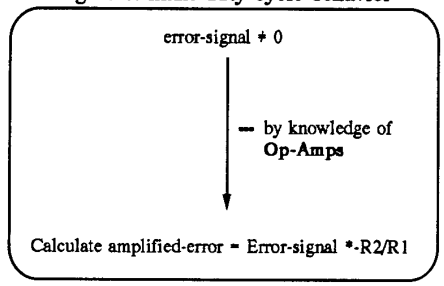

Figure 8: make-duty-cycle behavior

then our representation would have to be extended. As it stands now, we would be able to trouble shoot the ACC device to levels such as the OpAmp level (as indicated in Figure 8). Third, the chunks are organized about meaningful concepts: the known functions of the device. Fourth, the extensions which we have undertaken provide a highly organized, and meaningful way of capturing causal knowledge about the ACC device.

4.2. FM SIMULATION : ACC

Having determined the FM representation for the ACC, we can now use it to perform consequence finding given starting conditions, as outlined in Section 2.3. For simplicity all physical constants were given a value of 1. Since the behavior of the physical cruise control system is time-dependent, to simplify the simulation process, we assume that each pass in our simulation spans a fixed interval of time.

Reasoning Step 1: In this step, fix the initial conditions. Set the command speed to be 65 mph, and the vehicle speed to be 60 mph.

Reasoning Step 2: Index from the starting conditions to the behaviors that are applicable. The behaviors of Figure 4, Figure 6, and Figure 8 are each applicable. But on performing the appropriate "filtering" to obtain the highest level applicable behavior, only the behavior of Figure 4 remains.

Reasoning Step 3: Construct a *particularized state diagram* (PSD). This is a knowledge structure that represents the state changes the device will go through as a result of the stated initial conditions. The least level of detail in such a PSD corresponds to the behavior of Figure 4, and is shown in Figure 9. We obtain more levels of detail in the PSD by a process resembling macro-expansion of the link annotations in the original behaviors. Figure 10 shows the PSD at a greater level of detail. The nodes that are boxed in Figure 10 are those that also appear in Figure 9.

Reasoning Step 4: This step uses the PSD constructed in the previous step to determine what the consequences will be on the cruise control system as a result of the given starting conditions. The values of state variables are recorded in an auxiliary database. Before traversing the

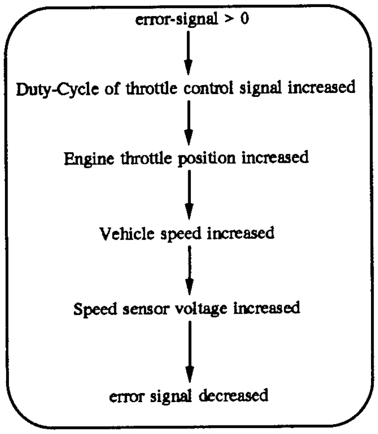

Figure 9: Lowest detail level of FM simulation on ACC

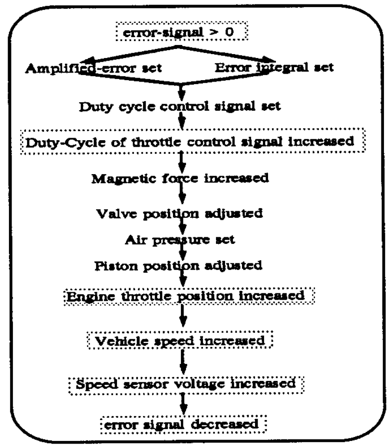

Figure 10: Detailed level of FM simulation on ACC

PSD, this database is initialized to the boundary conditions. Then as the PSD is traversed, appropriate changes are made to the state variable database as indicated by each node in the PSD. The values recorded in the database after one invocation of the FM simulator are shown in Figure 11.[2]

Note that the vehicle speed has increased from the initial condition (60 mph) and is approaching the set speed of 65 mph. Because the error signal is still not 0, the simulator will remain active; it will repeat the same steps as before, and will produce the values shown in Figure 12 at the end of the second invocation.

Although we not further discuss it here, it is interesting to note that after the third simulation (since the error signal is still not 0) the speed is over 65 mph. In fact, proceeding with the simulation produces a damped oscillating behavior. Although within our current framework the FM simulator could not recognize and label the behavior as "damped and oscillating," it was very encouraging to observe this result.

2. These values are not stored as the graph structure mirroring the PSD, but we show it that way here for pedagogical purposes.

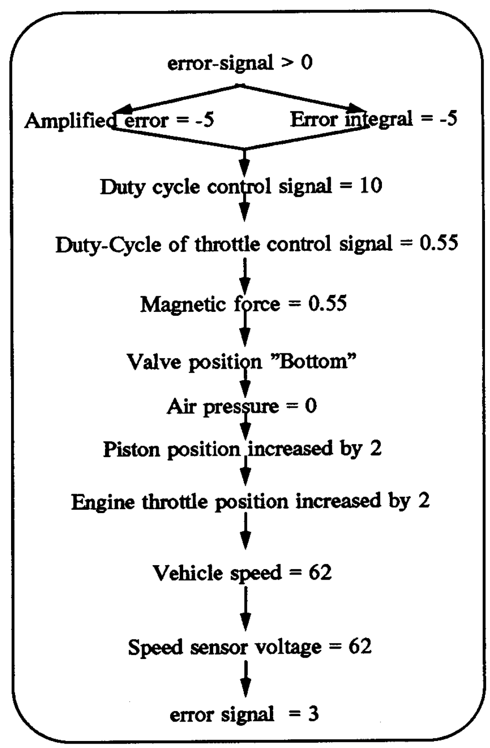

Figure 11: PSD showing state variable values after one pass

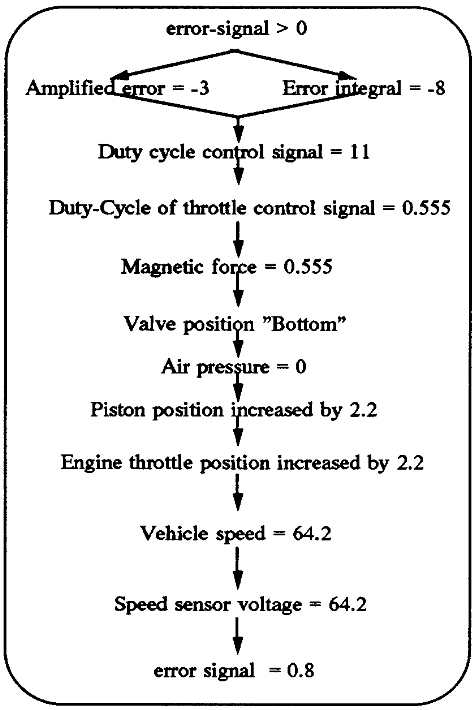

Figure 12: PSD showing state variable values after second pass

5. RELATED RESEARCH

The idea of integrating quantitative and qualitative models to reason about physical systems has recently attracted the attention of several researchers in the MBR community. While researchers in this area have differing flavors for the scheme they use for implementing the integration, the intuitions underlying their research and ours is common. That motivation lies in the inherent weakness of qualitative models: qualitative models cannot produce definitive predictions. On the other hand, typically qualitative models of a physical system typically yield more informative explanations of reasoning than do quantitative models. Finally, and of central importance, in many situations a combination of qualitative and quantitative relationships are known about a particular situation, and a modeler given such a situation would desire an integrated framework for representing all that is known about the target system - both quantitative and qualitative relationships.

In recent research by Berleant and Kuipers [20], a qualitative-quantitative simulator is built on the base of the QSIM approach. In this simulator, both qualitative and numeric state representations are maintained and used simultaneously during the simulation. In this way qualitative constraints can be cross-checked against their numeric counterparts, thus having the power of a numerical simulation without losing the flexibility of a qualitative simulation. In essence, the quantitative information alone is used for numerical simulation, while the qualitative information alone is used for qualitative simulation. While this research has the same motivation and goals as our research it differs in detail. In our work we do not maintain both quantitative and qualitative representations. Instead, we utilize one framework (the functional viewpoint) in our models which are represented partially numerically and partially qualitatively. We maintain a numeric representation for parts of the modeled system where complete knowledge is available and is useful. Meanwhile, we maintain a qualitative representation for other parts were complete knowledge is not available and/or not directly useful.

In other research by Forbus and Falkenhainer [21], a qualitative-quantitative integration was also undertaken, which in this case produces what are termed "self-explanatory simulations." This simulator is based on the constructs of the QP theory. Forbus and Falkenhainer use a qualitative domain model to produce a total envisionment for the physical system. A math-model library is then used to construct a set of ordinary differential equations for each qualitatively distinct region of behavior identified in the envisionment. A simulation procedure is then written for each set of equations based on the state transitions in the envisionment. These procedures collectively form a simulator for the physical system. By creating a math-model library, they automated the process of constructing the differential equations, as well as the generation of the simulator. While this removes the burden of generating simulators, some detail will typically be lost, thus losing the power and accuracy of a numerical simulation. Our work differs with Forbus and Falkenhainer at this point. Since we manually generate the model, we will more usually have a simulator which is as accurate as the model. On the other hand, the automatic generation of a model is a goal which is quite laudable in and of itself.

6. CONCLUSIONS

In his survey of Model Based Reasoning, Davis lists three crux research issues that MBR approaches must deal with: issues of domain independence, issues of scalability, and issues of model selection . Although the review is explicitly over the area of troubleshooting, the same three issues may be raised for the entire area of MBR.

The issue of domain independence is of whether a particular MBR technique is applicable to only a limited domain, or if it is more generally useful. In this report we have outlined our experience on one research project, a testbed project to apply FM to the cruise control problem. To facilitate a solution, we eventually developed new primitives for our languages in which FM systems are written to include parametrized state variables. This simple extension yielded all the power necessary to then use the extended FM approach as a template for organizing a series of numerical calculations about a physical device. The extensions we made were not specific for the ACC, but would rather be applicable over a wide variety of devices. Going research listed here, we have work in progress at Michigan State to apply the Functional Modeling approach to modeling problem of global ecological cycles, mixed systems in high performance aircraft[3], composite materials, and business organizations. As results from these projects accumulate, we expect to gain more confidence in the generality of the FM.

The issue that Davis raises of scalability is a central concern for MBR approaches. One way to argue for a scalable approach is to point out ways in which the approach modularizes a domain. The Functional Approach deals with this issue very directly; behaviors in the Functional Approach are causal net *fragments*. The organization of the fragments is by the known functionality of the device that is being modeled. Because a Functional Representation of a device is inherently compartmentalized, it is easy from a representation viewpoint to add new subdevices.

We have shown that with proper extension, we can utilize FM as a framework for organizing numerical calculations about a device. An apt question would be "So what, the numerical calculations about the ACC could have been expressed easily in a simple PASCAL program?" One reply to that sort of question is that yes, all the numerical calculations required to carry out the solution of the ACC *could* have been done in PASCAL, but how well would such an approach *scale*? Suppose we were representing a nuclear power plant. How difficult would it be to develop a similar PASCAL program in that case? The reason that our FM approach to organizing numerical calculations will scale well is that the approach emphasizes (a) modularity, and (b) an organization based around known functionality of the device. In our approach, the numerical calculations are simply ways of determining the results of changes in state that are necessary to achieve known functionality. (E.g., see Figure 6 on page 5.)

The final issue of Davis, model selection, is both the most interesting of his three issues, and the hardest to pin down. One of the reasons for the slipperiness is that selection of model is a multidimensional task. Along one dimension, we must select the *level* at which we want to represent our model. As Davis points out, no model is complete. The Functional Representation deals straightforwardly with this fact by including the ability to point to "world knowledge" as the reason for a given state variable transition (in a behavior). This gives an ability to the modeler to construct a model that "bottoms out" at whatever level is appropriate.[4]

The issue of the type of model we want to construct should be based on (a) the representational primitives offered by a particular type of model, and (b) the reasoning that a particular type of model enables. If the knowledge we have of a device to be modeled can be expressed in the primitives of a particular approach, and if the output of reasoning with that approach matches what we need to have in terms of output, then that particular type of modeling approach would be a good candidate. This statement may seem self-evident. Yet for the most part, MBR has not dealt explicitly with issues of *types* of models in these terms. We believe that one of the strongest arguments supporting the FM approach to MBR is the relative clarity of statement of the representational primitives of the approach, and of the reasoning methods that come bundled

3. By "mixed systems" we mean systems that have component subsystems which are built emphasizing different physical properties. E.g., an actuator system that includes electrical analog units, electrical digital units, hydraulic units, and mechanical units.

4. The level at which the bottoming is legitimate is determined by whether or not the world knowledge can be treated as a monolithic entity for purposes of the current model. A full discussion of this issue is beyond the scope of this paper.

with the approach.

Finally, we emphasize the importance we attach to identifying good testbed problems. Such problems are forceful because we explore them to exercise our current theoretical stances, and do so in reasonable time. The extensions we made to the FM approach were made in direct response to considering the cruise control problem. Moreover, once the necessary extensions were completed, we gained new insight into the functional approach itself. Incremental xtension of existing theory as new problems are encountered and solved is what Thomas Kuhn calls "normal science" [22]. Its practice is usually taken to be a sign that a scientific field is maturing.

7. ACKNOWLEDGMENTS

Research reported here benefitted from discussion with Juemin Sun, Mahmoud Pegah, and Gene Wallingford. In addition, participants in the Workshop for Functional Reasoning, held at Ohio State University in April, 1990, made many useful comments. Finally, Chris Price is to be congratulated for giving the community an interesting test bed problem.

Research reported here was supported in part by the McDonnell Douglas Independent Research and Development Program, by NSF Grant #9009014, by the Ameritech Corporation, by the Research Excellence Fund - State of Michigan, and by generous equipment support from Apple Computer.

8. REFERENCES

[1] Genesereth, M.R. (1984) The use of design descriptions in automated diagnosis. Artificial Intelligence 24: 411-436.

[2] Davis, R. (1984) Diagnostic reasoning based on structure and behavior. Artificial Intelligence 24: 347-410.

[3] deKleer, J. & Williams, B. (1987) Diagnosing multiple faults. Artificial Intelligence 32: 97-130.

[4] Patil, R., Szolovitis, P. & Schwartz, W. (1981) Causal understanding of patient illness in medical diagnosis. In Proceedings of IJCAI-81. 893-899.

[5] Pan, J. (1984) Qualitative reasoning with deep-level mechanisms models for diagnosis of mechanism failures. In Proceedings of CAIA-84. 295-301.

[6] Forbus, K. D. (1984). Qualitative Process Theory. Artificial Intelligence. 24 (pp. 85-168).

[7] D'Ambrosio, B. (1989). Extending the Mathematics in Qualitative Process Theory. In Daniel S. Weld and J. deKleer, Readings in Qualitative Reasoning about Physical Systems. (p 133, Chapter 5).

[8] Chiu, C. (1989). Constructing Qualitative Domain Maps from Quantitative Simulation Models. In L. E. Widman, K. A. Loparo, & N. R. Nielsen, Artificial Intelligence, Simulation and Modeling. (pp. 275-299, Chapter 11). John Wiley and Sons.

[9] Karp, P., & Friedland, P. (1989). Coordinating the Use of Qualitative and Quantitative Knowledge in Declarative Device Modeling. In L. E. Widman, K. A. Loparo, & N. R. Nielsen, Artificial Intelligence, Simulation and Modeling. (pp. 189-207, Chapter 7). John Wiley and Sons.

[10] Sembugamoorthy, V., & Chandrasekaran, B. (1986). Functional Representation of Devices and Compilation of Diagnostic Problem-Solving Systems. In J. Kolodner, & C. Reisbeck (ed), Experience, Memory, and Learning. Lawrence Erlbaum Associates.

[11] Sticklen, Jon, & Chandrasekaran, B. (1985). Use Of Deep Level Reasoning in Medical Diagnosis. In Proc. of The Expert Systems in Government Symposium. McLean, Virginia.

[12] Sticklen, J. (1987). MDX2: An Integrated Medical Diagnostic System. PhD dissertation. The Ohio State University. Columbus, Ohio.

[13] Sticklen, J., & Chandrasekaran, B. (1989). Integrating Classification-Based Compiled Level Reasoning with Function-Based Deep Level Reasoning. Applied Artificial Intelligence. 3 (pp. 275-304).

[14] Goel, A., & Chandrasekaran, B. (1989). Functional Representation of Designs and Redesign Problem Solving. Proceedings of IJCAI-89.

[15] Punch, W. F. (1989). A Diagnosis System Using a Task Integrated Problem Solving Architecture (TIPS), Including Causal Reasoning. The Ohio State University. Columbus, Ohio.

[16] Allemang, D. (1990). Understanding Programs as Devices. PhD dissertation. The Ohio State University. Columbus, Ohio.

[17] Keuneke, A. (1989). Machine Understanding of Devices: Causal Explanation of Diagnostic Conclusions. PhD dissertation. The Ohio State University. Columbus, Ohio.

[18] Chittaro, L., Costantini, C., Giovanni, G., Tasso, C., & Toppano, E. (1989). Diagnosis based on cooperation of multiple knowledge sources. In Proceedings of the Model Based Diagnosis International Workshop. Paris.

[19] Franke, D. W. (1989). Representing and Acquiring Teleological Descriptions. In Proceedings of the 1989 Workshop on Model Based Reasoning (IJCAI 89). Detroit. (pp. 62-67).

[20] Berleant, Daniel & Kuipers, Benjamin. (1990) Qualitative-Quantitative Simulation with Q3. Proceedings of the Second AAAI Workshop on Model Based Reasoning: 72-81.

[21] Forbus, Kenneth D. & Falkenhainer, Brian. (1990) Self-Explanatory Simulations: An Integration of qualitative and quantitative knowledge. Proceedings of AAAI-90: 380-387.

[22] Kuhn, T. S. (1970). The Structure of Scientific Revolutions. University of Chicago Press.