N94-34041

The Stanford How Things Work Project[1]

Richard Fikes, Tom Gruber, and Yumi Iwasaki

Knowledge Systems Laboratory
Stanford University
701 Welch Road, Building C
Palo Alto, CA 94304

Abstract

We provide an overview of the Stanford How Things Work (HTW) project, an ongoing integrated collection of research activities in the Knowledge Systems Laboratory at Stanford University. The project is developing technology for representing knowledge about engineered devices in a form that enables the knowledge to be used in multiple systems for multiple reasoning tasks, and reasoning methods that enable the represented knowledge to be effectively applied to the performance of the core engineering task of simulating and analyzing device behavior. The central new capabilities currently being developed in the project are automated assistance with model formulation and with verification that a design for an electro-mechanical device satisfies its functional specification.

Introduction

The Stanford How Things Work Project is an ongoing integrated collection of research activities in the Knowledge Systems Laboratory at Stanford University [Fikes, et al 91] led by Richard Fikes. The overall objective of the project is to develop knowledge-based technology that will enable computer systems to offer intellectual assistance at high levels of competence to problem solvers and decision makers in all stages of the life cycle of engineered products. To achieve that objective, we are developing:

- Technology for representing knowledge about engineered devices in a form that enables the knowledge to be used in multiple systems for multiple reasoning tasks, and
- Reasoning methods that enable the represented knowledge to be effectively applied to the performance of the core engineering task of simulating and analyzing device behavior.

The knowledge to be represented includes a broad range of subjects, from the fundamentals of physics and engineering, to device models that describe device structure, behavior, and function, to the rationale for the design of specific devices. In order to directly support the reuse of encoded engineering knowledge bases, we are working with other research groups to establish a common device modeling language and a clearing house for device models. The common language will be based on and make use of the languages and tools being developed in the DARPA Knowledge-Sharing Initiative [Neches, et al 91].

[1] This research sponsored by DARPA and NASA under NASA grants NAG 2-581 and NCC 2-537.

The primary engineering task on which we are focusing is that of supporting the design of electromechanical devices by providing effective tools for *simulating and analyzing the behavior of such devices* in all stages of their design. Simulation technology has the potential of providing a rapid low-cost means of testing new designs for sophisticated equipment in many industries before acquisition decisions are made and expensive prototypes are built. In order to realize that potential, simulators need to have three key capabilities they are currently lacking. Namely, simulators need to be:

- **Applicable to partially specified designs** -- Many of the financially significant decisions about new designs are made during conceptual and preliminary design stages. *Qualitative* simulation techniques are needed to obtain behavior analyses during those early stages of design, since the detailed design specifics required to do conventional numerical simulations are not yet available.
- **Rapidly reconfigurable** -- Simulators need to be capable of supporting a broad range of tests of a new design that include variations in level of detail (from engineering analyses of individual subsystems to macro level mission effectiveness evaluations of the overall design), issues being addressed (fuel consumption rates, ease of operation, response speed, etc.), and operating conditions being considered (extreme weather, variations in operator training, etc.). No one simulator or one simulation model will be able to support such a range of tests. Thus, designers need to be provided with a *simulation foundry* that enables them to rapidly configure and run multiple simulations on an as-needed basis to answer specific analysis questions.
- **Self interpreting** -- In order for designers to use simulators for multiple purposes on a routine basis, simulation results must be understandable with minimum effort. Thus, simulators need to provide significant assistance with the task of interpreting their output by producing summaries, explanations, and analyses which are directly oriented to a given analysis task.

We are developing knowledge-based technology that will remove those deficiencies. That is, we are developing augmentations to conventional simulators that will enable them to become applicable to partially specified designs, rapidly reconfigurable, and self-interpreting. In particular, we are developing techniques for:

- Automatically formulating a simulation model that embodies the abstractions, approximations, assumptions, and perspectives that are appropriate for a given analysis task,
- Performing qualitative simulation of device modules which have not yet been designed in detail or whose detailed quantitative behavior is not relevant to a given analysis task,
- Automatically guiding a simulator to consider scenarios that are relevant to a given analysis task,
- Generating human-understandable *causal* explanations of simulation results,
- Automatically determining whether simulated behavior satisfies functional specifications, and
- Testing and automatically generating procedures for operating the device.

We are embodying the techniques developed in our research in an evolving prototype "designer's associate" system called the *Device Modeling Environment (DME)* [Iwasaki and Low 91]. The DME system is intended to be useful to the research community at large as an experimental testbed, educational tool, and foundation on which to build new representation and reasoning capabilities. DME has already been developed to a

sufficient level of maturity to provide both a demonstration vehicle and a useful experimental testbed within the project.

DME is intended to enable a designer to document a design as it evolves and to support experimentation with alternative designs. The current system is used as follows:

- **Designer describes device** -- The designer selects components from a library and specifies the structural connections among the components.
- **Designer selects behavior models** -- The designer selects from a library the models of component behavior that provide the abstractions, approximations, assumptions, and perspectives which are appropriate for the analysis he or she wants to do.
- **DME generates simulation model** -- DME uses the device model specified by the designer to generate a qualitative *or* quantitative simulation model of the device .
- **Designer interactively guides the simulation** -- The designer uses a simulator provided by DME to interactively explore possible device behaviors .
- **DME provides causal explanations of simulated behavior**.
- **Designer analyzes behavior** -- The designer compares the predicted behavior with the intended functionality of the device.

New Capabilities Being Produced

The current DME system embodies state of the art research results. It provides an integrated set of tools for performing what might be characterized as a limited form of semi-automatic behavior analysis. For example, the system automatically formulates a simulation model, but only after the designer has selected from the system's model library appropriate behavior models for each device component.

Our current research is focused on taking steps toward providing a designer with a *comprehensive* and *fully automated* behavior analysis of a device being designed. Our three year goal is to develop new capabilities and integrate them into DME so that the system could be used as follows:

- **Designer describes device** -- In addition to the current facilities for selecting components from a library and specifying the structural connections among the components, new facilities will be developed to enable the designer to specify:
  - Intended functionality of a device,
  - Expected operator interactions with a device,
  - Assumptions about the environment in which a device will be operating, and
  - Rationale for design decisions;
- **DME formulates appropriate behavior model** -- New facilities will be developed that will enable DME to determine the abstractions, approximations, assumptions, and perspectives that are appropriate for specific analysis tasks such as testing whether the device design satisfies the functional specifications.
- **DME generates appropriate simulation model** -- DME will use the structural and behavioral device models to generate a simulation model of the device that *intermixes* qualitative and quantitative simulation as needed. New facilities will be developed to enable it to select an appropriate qualitative or quantitative simulator for each device

module depending on the level of detail at which the module has been designed and the level of detail required by the analysis task.

- **DME guides the simulation** -- New facilities will be developed to enable DME to direct the simulator to consider scenarios that are relevant to a given analysis goal such as testing whether the functional specification is satisfied.
- **DME determines whether behavior achieves the intended functionality** – New facilities will be developed to enable DME to compare the simulated behavior with the functional specification. In cases where the behavior does not satisfy the specifications, DME will be able to provide feedback in the form of additional constraints on the design which would guarantee that the device behaves as intended.
- **DME explains behavior analysis results** -- In addition to the current facilities for providing causal explanations of simulated behavior, new facilities will be developed to explain how and why the design either does or does not satisfy the functional specification.

An additional significant capability being developed in our project which is not highlighted in the above scenario is the use of DME for designing and analyzing procedures for operating a device. For example, DME seems particularly useful for assisting with the verification of procedures that respond to device malfunctions in that it enables simulation models to be rapidly reformulated to reflect malfunctions and can explain the effects caused by the procedures. We are currently working with NASA on such a procedure verification application in which DME will be used for both procedure debugging and operator training.

The central new capabilities currently being developed for DME are automated assistance with model formulation, automated assistance with verification that a design satisfies its functional specification, and automatic generation of causal explanations of device behavior. Our approach to achieving these capabilities is summarized in the sections below.

Automatic Model Formulation

We are developing methods for providing automated assistance with the core problem of model formulation -- a service that will help engineers build nontrivial models of device behavior for specific purposes.

The state of the art in model formulation today is model configuration from libraries of component models. Simulators such as SPICE [Katzenelson 66] and those for VHDL [Harr and Stanculescu 89] are based on libraries of component models which have been preformulated by modeling experts. The user selects components and configures them, and then the system compiles the code necessary to run a simulation and plot the trajectories of variables.

Today's component-based model libraries are most successful in those domains where components are well-defined idealizations at a single level of abstraction, such as logical circuits. The mapping between physical components and idealized component models is simple, and there is exactly one behavior model associated with each component model. Thus, the engineer's part of the model formulation task is simplified in that he or she need only specify a component connection topology.

However, in most domains and tasks, the mapping from phenomena of interest in a physical system to a set of possible behavior models is complex and the result of nontrivial

reasoning. In doing model formulation, an engineer must identify the *relevant abstractions* to model, deciding, for example, whether to treat the load of an electrical power supply as a single resistive element or as a system with components that vary in their power usage. The engineer must also make *simplifying assumptions* and *approximations*, such as to assume no friction in a gear train or that a valve can be modeled as a discrete switch. The engineer makes these modeling choices to produce a model which answers a particular information need in a reasonable amount of time.

The power of the library approach derives from the reuse of the component models and the automatic assembly of system models from partial descriptions. DME will achieve the same advantages of knowledge reuse and automation, but for a more general class of domains and for multiple modeling purposes.

Even in domains such as analog circuits where there is a large library of ready-made simulation modules for standard components, building a model of an entire system is not an easy task. There are typically many simulation modules for each type of component, each based on different simplifying assumptions and approximations which are not stated explicitly. Therefore, a significant amount of effort and expertise is required for engineers to use even off-the-shelf simulation modules to assemble a model of a whole system. Engineers often prefer to write their own modules instead of using off-the-shelf modules precisely because they do not know all the underlying assumptions and do not trust their results.

DME will enable *knowledge reuse* by providing the representation and architecture for model libraries containing a comprehensive body of behavior model fragments, each making particular abstractions and approximations and conditioned on *explicitly represented* modeling assumptions. The formalism and examples will allow engineers to fill the libraries with model fragments covering those phenomena they need to model. We expect that a market will develop for these models, possibly driving a small industry of component-model-building specialists (as in electronics).

DME will provide *automated model formulation assistance* using these libraries. The assistance will change the nature of the interaction between the human engineer and the computational environment. Instead of operating at the level of equations or fixed-level component models, the engineer may specify the high-level device structure, the simulation goal, the utility criteria, a description of the context of use, and any initial conditions. The system will take an active role in selecting appropriate model fragments to construct a complete and coherent simulation model. This advance in model formulation is similar to the improvement in software development from early assembly-language programming to Fourth Generation Language environments.

Automatic Behavior Verification

Understanding the design of an engineered device requires both knowledge of the general physical principles that determine the behavior of the device and knowledge of what the device is intended to do (i.e., its functional specification). However, the majority of work in model-based reasoning about device behavior has focused on modeling a device in terms of general physical principles *or* intended functionality, but not both. For example, most of the work in qualitative physics has been concerned with predicting the behavior of a device given its physical structure and knowledge of general physical principles. In that work, great importance has been placed on preventing a pre-conceived notion of an intended function of the device from influencing the system's reasoning methods and representation of physical principles in order to guarantee a high level of "objective truth"

in the predicted behavior. In contrast, in their work based on the FR (Functional Representation) language [Sembugamoorthy & Chandrasekaran, 1986] [Keuneke, 1991], Chandrasekaran and his colleagues have focused mostly on modeling a device in terms of what the device is intended to do and how those intentions are to be accomplished through causal interactions among components of the device.

Both types of knowledge, functional and behavioral, seem to be indispensable in fully understanding a device design. On the one hand, knowledge of intended function alone does not enable one to reason about what a device might do when it is placed in an unexpected condition or to infer the behavior of an unfamiliar device from its structure. On the other hand, knowledge of device structure and general physical principles may allow one to predict how the device will behave under a given condition, but without knowledge of the intended functions, it is impossible to determine if the predicted behavior is a desirable one, or what aspect of the behavior is significant.

In order to use both functional and behavioral knowledge in understanding a device design, it is crucial that the functional knowledge is represented in such a way that it has a clear interpretation in terms of actual behavior. Suppose, for example, that the function of a charge current controller is to prevent damage to a battery by cutting off the charge current when the battery is fully charged. To be able to determine whether this function is actually accomplished by an observed behavior of the device, the representation of the function must specify conditions that can be evaluated against the behavior. Such conditions might include occurrence of a temporal sequence of expected events and causal relations among the events and the components. Without a clear semantics given to a representation of functions in terms of actual behavior, it would be impossible to evaluate a design based on its predicted behavior and intended functions.

While it is important for a functional specification to have a clear interpretation in terms of actual behavior, it is also desirable for the language for specifying functions to be independent of any particular system used for simulation. Though there are a number of alternative methods for predicting behavior, such as numerical simulation with discrete time steps or qualitative simulation, a functional specification at some abstract level should be intuitively understandable without specifying a particular simulation mechanism. If a functional specification language was dependent on a specific simulation language or mechanism, a separate functional specification language would be needed for each different simulation language, which is clearly undesirable. What is needed is a functional specification language that has sufficient expressive power to support descriptions of the desired functions of a variety of devices. At the same time, the language should be clear enough so that for each simulation mechanism used, it can be given an unambiguous interpretation in terms of a simulated behavior.

An essential element in the description of a function is causality. In order to say that a device has achieved a function, which may be expressed as a condition on the state of the world, one must show not only that the condition is satisfied but also that the device has participated in the causal process that has brought about the condition. For example, when an engineer designs a thermostat to keep room temperature constant, the design embodies her idea about how the device is to work. In fact, the essential part of her knowledge of its function is the expected causal chain of events in which it will take part in achieving the goal. Thus, a representation formalism of functions must provide a means of expressing knowledge about such causal processes.

We are developing a new representational formalism for specifying device functions called CFRL (Causal Functional Representation Language) that allows functions to be expressed in terms of expected causal chains of events [Vescovi, Iwasaki, Fikes, &

Chandrasekaran, 1993]. We are providing the language with a well-defined semantics in terms of the type of behavior representation widely used in model-based, qualitative simulation. Finally, we are using CFRL as the basis for a function verification program which determines whether a behavior achieves an intended function.

Explanation Generation

We are developing a method for generating explanations of how devices work and incorporating that method in DME [Gruber & Gautier, 1992; Gruber & Gautier, 1993]. On the basis of an initial device model and the behavioral predictions obtained through simulation, DME can answer a range of user queries about the structure and behavior of the modeled system.

The approach we are developing combines several techniques for explanation generation:

- Automatically synthesizing formal mathematical models from high-level model specifications, and explaining low-level simulation data in terms of the original specifications
- Inferring causality from mathematical models, rather than assuming ad hoc, hand-crafted causal models
- Dynamically generating explanations in response to user queries, which are formulated by direct manipulation on text and graphics displayed during simulation
- Supporting interactive follow-up questions, allowing the user to get more information on a particular point of an explanation
- Using a compositional method of text generation, in which textual annotations of model fragments are composed into phrases, which are then processed to produce smooth, concise text.

The explanations are intended for three application tasks: data interpretation, the design of operator procedures, and design documentation. The task of interpreting simulation data is important for exploring hypotheses about device behavior during conceptual design and for debugging the simulation model itself. Machine-generated explanations can facilitate data interpretation by showing the relationship between low-level simulation data and the original modeling decisions and assumptions. In the second application, operators of equipment need to rapidly explore failure scenarios in order to design and test corrective procedures. Dynamically generated causal explanations can help the operator assess the situation and determine appropriate actions. Finally, self-explanatory simulations can be used to document design intent [Gruber, 1990; Gruber, 1991]. Instead of writing a static design document that is often inaccurate and out of date, the designer can demonstrate the intended and expected behavior of a device using simulation. The system can generate explanations in response to questions by the reader.

An important element of the explanation approach in DME is the use of real engineering models, rather than ad hoc "causal models" that are built specifically for explanation generation. In explaining how things work, people *do* use causal terminology. However, when analyzing the behavior of devices, engineers use formalisms such as logical and mathematical constraints that are not causal. DME *infers causal dependencies* among modeled parameters by analyzing logical and mathematical constraints.

In DME, logical constraints occur in the preconditions of model fragments. Discrete events, such as changes in the operating regions of components or discontinuous changes in quantitative parameters, are due to changes in the activation of model fragments. The

"cause" of a discrete event, then, can be viewed as the set of facts and parameter values that satisfied the preconditions of a model fragment representing the event. DME can therefore explain the cause of discrete events by describing how the preconditions of model fragments are satisfied. It can then recursively explain the causal ancestry of each of the facts or variable values in the preconditions. This is similar to the traditional approach to explaining rule firings in expert systems. In DME, heuristics are applied to filter some of the facts and variables, producing a more concise explanation.

The collection of techniques we are developing constitute a practical method for generating interactive explanations of device behavior in natural language. No special knowledge of linguistics is needed for building models; the engineer merely annotates behavior models developed for simulation. Because causal relationships are inferred for each simulation scenario, there is no need to build in assumptions of causality in the models. The modeling and simulation technology that underlies the approach is realistic for physical systems that can be modeled with time-varying ordinary differential equations, such as electromechanical devices for controlling position or force (e.g., robot manipulators), and process control systems (e.g., control of fuel supply for the Space Shuttle).

Bibliography

[Cutkosky et al 93] Cutkosky, M. R., Engelmore, R. S., Fikes, R. E., Genesereth, M. R., Gruber, T. R., Mark, W. S., Tenenbaum, J. M., & Weber, J. C. (1993). PACT: An Experiment in Integrating Concurrent Engineering Systems. IEEE Computer, 26(1), pp. 28-37. Also available as KSL 93-21.

[Fikes, et al 91] Richard Fikes, Tom Gruber, Yumi Iwasaki, Alon Levy, and Pandu Nayak; "How Things Work Project Overview"; Knowledge Systems Laboratory Technical Report KSL 91-70; Computer Science Department, Stanford University; November 1991.

[Gautier & Gruber 93] Gautier, P. O. & Gruber, T. R. (1993). Generating Explanations of Device Behavior Using Compositional Modeling and Causal Ordering. Eleventh National Conference on Artificial Intelligence. July 1993. AAAI Press. Also available as KSL 93-06.

[Gruber 92] Gruber, T. (1992). Model Formulation as a Problem Solving Task: Computer-assisted Engineering Modeling. International Journal of Intelligent Systems, 8(1) , 105-127. KSL 92-57.

[Gruber & Gautier 93] Gruber, T. & Gautier, P. (1992). Machine-generated Explanations of Engineering Models: A compositional modeling approach. To appear in the Proceedings of the 1993 International Joint Conference on Artificial Intelligence. August 1993. Also available as KSL 92-85.

[Gruber and Russell 92] Gruber, T. & Russell, D. M. (1992). Derivation and Use of Design Rationale Information as Expressed by Designers. Also available as KSL 92-64.

[Gruber and Russell 92a] Gruber, T. & Russell, D. M. (1992). Generative Design Rationale: Beyond the Record and Replay Paradigm. Collection on Design Rationale. Kluwer Academic Publishers. Also available as KSL 92-59.

[Gruber 92a] Gruber, T. R. (1992). A Translation Approach to Portable Ontology Specifications. Second Japanese Knowledge Acquisition Workshop, Kobe and Tokyo, Japan. Also available as KSL 92-71.

[Gruber 93] Gruber, T. R. (1993). Toward Principles for the Design of Ontologies Used for Knowledge Sharing. KSL 93-04.

[Harr and Stanculescu 89] R. Harr and A. Stanculescu (Eds.); *Applications of VHDL to Circuit Design*; Kluwer Academic Publishers; Norwell, Massachusetts; 1989.

[Iwasaki et al 93] Iwasaki, Y., Fikes, R., Vescovi, M., & Chandrasekaran, B. (1993). How Things are Intended to Work: Capturing Functional Knowledge in Device Design. To appear in the Proceedings of the 1993 International Joint Conference on Artificial Intelligence. August 1993. Also available as KSL 93-39.

[Iwasaki & Levy 93] Iwasaki, Y. & Levy, A. Y. (1993). Automated Model Selection for Simulation. QR93. Also available as KSL 93-11.

[Iwasaki and Chandrasekaran 92] Y. Iwasaki and B. Chandrasekaran; "Design Verification Through Function and Behavior-Oriented Representations: Bridging the Gap Between Function and Behavior"; Proceedings of the Second International Conference On Artificial Intelligence in Design; 1992.

[Iwasaki and Low 91] Y. Iwasaki and C. M. Low; "Model Generation and Simulation of Device Behavior with Continuous and Discrete Changes: the Details"; Technical Report, KSL 91-30, Knowledge Systems Laboratory, Stanford University, 1991.

[Katzenelson 66] J. Katzenelson; "AEDNET: A simulator for nonlinear networks"; Proceedings of the IEEE, 54(11).

[Levy, et al 92] Levy, A. Y., Iwasaki, Y., & Motoda, H. (1992). Relevance Reasoning to Guide Compositional Modeling. Second Pacific Rim International Conference on Artificial Intelligence, Seoul, Korea. Also available as KSL 92-47.

[Levy et al 93] Levy, A. Y., Mumick, I. S., Sagiv, Y., & Shmueli, O. (1993). Equivalence, Query-reachability and Satisfiability in Datalog Extensions. Twelfth ACM-SIGACT-SIGART-SIGMOD Symposium on Principles of Database System (PODS-93). Also available as KSL 93-15.

[Levy & Sagiv 93] Levy, A. Y. & Sagiv, Y. (1993). Controlling Inference Using the Query Tree. Biannual Israeli Symposium on Foundations of AI-93. Also available as KSL 93-07.

[Levy & Sagiv 93a] Levy, A. Y. & Sagiv, Y. (1993). Exploiting Irrelevance-reasoning to Guide Problem Solving. To appear in the Proceedings of the 1993 International Joint Conference on Artificial Intelligence. August 1993. Also available as KSL 93-05.

[Levy & Sagiv 93b] Levy, A. Y. & Sagiv, Y. (1993). Queries Independent of Updates. Twelfth ACM-SIGACT-SIGART-SIGMOD Symposium on Principles of Database System (PODS-93). Also available as KSL 93-03.

[Nayak 92] Nayak, P.P. Automated Modeling of Physical Systems. (Ph.D. Dissertation) September 1992.

[Nayak 92a] Nayak, P. P. Order of Magnitude Reasoning Using Logarithms. Third International Conference on the Principles of Knowledge Representation and Reasoning. October 1992. Morgan Kaufman. Also available as KSL 92-29.

[Nayak 92b] Nayak, P. P. Causal Approximations. Tenth National Conference on Artificial Intelligence. July 1992. AAAI Press. Pgs. 703-709.

[Nayak et al 92] Nayak, P. P., Joskowicz, L., Addanki, S. Automated Model Selection using Context-Dependent Behaviors. Tenth National Conference on Artificial Intelligence. July 1992. AAAI Press. Pgs. 710-716.

[Neches, et al 91] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, & W. Swartout; "Enabling Technology for Knowledge Sharing"; *AI Magazine*, vol. 12, no. 3, pp 16-36; 1991.

[Patil et al 92] Patil, R. S., Fikes, R. E., Patel-Schneider, P. F., Mckay, D., Finin, T., Gruber, T. R., & Neches, R. (1992). The DARPA Knowledge Sharing Effort: Progress Report. Principles of Knowledge Representation and Reasoning, Cambridge, MA, 777-788, Morgan Kaufmann. Also available as KSL 93-23.

[Sembugamoorthy & Chandrasekaran 86] V. Sembugamoorthy and B. Chandrasekaran; "Functional Representation of Devices and Compilation of Diagnostic Problem-Solving Systems"; in J. L. Kolodner and C. K. Riesbeck (editors); *Experience, Memory, and Reasoning*; Lawrence Erlbaum Associates, Hillsdale, NJ; 1986.

[Vescovi, et al 93] Vescovi, M., Iwasaki, Y., Fikes, R., & Chandrasekaran, B. (1993). CFRL: A Language for Specifying the Causal Functionality of Engineering Devices. Eleventh National Conference on Artificial Intelligence. July 1993. AAAI Press. Also available as KSL 93-38.