



SmartVehicle Evaluation Report

Modeling in Hybrid System Interchange
Format

In support of the University of California Open
Experimental Platform for DARPA – MoBIES,
Contract F33615-01-C-1841.

Authors:

Shiva Sivashankar
Emmeskay, Inc.
Suite 300, 44191 Plymouth Oaks Blvd.
Plymouth, MI 48170
shiva@emmeskay.com

William P. Milam(Contact)
Ford Research Laboratory
Dearborn, MI
wmilam@ford.com

1. Introduction

One of the goals of the Ford MoBIES group has been to set up realistic automotive benchmark problems and evaluate the Hybrid System Tools proposed by the various MoBIES research groups. Three research tools, henceforth called Hybrid System Tools, have been identified for such an evaluation – CheckMate from Carnegie Mellon University, Charon from University of Pennsylvania and SAL from SRI. As part of this effort, the Ford MoBIES group performed a benchmark-based evaluation of CheckMate and documented the results of this evaluation in a baseline report [1]. During ongoing discussions between the MoBIES researchers, it became obvious that the three tools were incompatible in the sense that the models and analysis performed in one tool cannot be automatically carried over to another tool. The lack of an automatic model conversion tool was seen as a major drawback. Working with other researchers at Vanderbilt University, the MoBIES Hybrid System Tools researchers came up with the idea of sharing model information between the tools using the Hybrid System Interchange Format (or HSIF for short). Using the semantics described by HSIF, each of the Hybrid System tools could export the model information into this common format. The other Hybrid System tools could use a corresponding import feature to read this model created in the original tool. This would avoid the tedious process of creating and maintaining model exchange processes between the individual tools. This could also result in the development of a “universal” tool-independent common format to represent Hybrid Systems. A first version of the HSIF semantics was developed at the beginning of this year [2]. A syntax document was also released based on these initial semantics [3]. This report is based on these versions of the HSIF semantics and syntax. The original HSIF semantics and syntax have been updated many times. The updated version of the HSIF semantics can be found in [4].

Instead of individually creating the automotive benchmark problem in each of the MoBIES Hybrid System tools, the Ford MoBIES group decided to create the benchmark example in HSIF. The Hybrid System Tools researchers and the researchers at Vanderbilt University are creating import and export programs between HSIF and the individual tools. When the automotive benchmark example is available in HSIF, one could leverage these programs to directly import the model into the individual tools. In addition, the process of creating the benchmark example in HSIF would highlight the utility of the existing HSIF semantics and would also show some deficiencies (if any) in the existing semantics.

This report describes the process and the results of creating the benchmark example in HSIF. The original benchmark example in Simulink is briefly described in the next section. This is followed by a discussion of the HSIF tool and the manual model creation process of the benchmark model. The process of translating an HSIF model into one of the Hybrid System Tools using an automated translator is also discussed here. In Section 5, some observations are made on the semantics of HSIF based on this model creation exercise. Finally, some concluding remarks are presented.

2. Benchmark Model in Simulink

A benchmark automotive Hybrid System example has been created by the MoBIES group participating from University of California at Berkeley [5]. This example model has been used for other tool evaluations in the past [1]. The transmission shift dynamics and the transmission shift controller form the core of this benchmark problem [6]. This model was substantially simplified to evaluate CheckMate. This simplified closed loop model (henceforth referred to as SCLM) reduced the engine to a look-up table and the vehicle longitudinal dynamics to a single state. A description of the simplified model and the Hybrid System analysis problem posed on this simplified problem can be found in [1].

When we started working on HSIF examples, it became clear to us that implementing SCLM in HSIF might be over-ambitious. Instead, we decided to first convert the transmission controller in SCLM into HSIF and then pursue the goal of converting the complete SCLM into HSIF at a later stage. This transmission controller in Simulink is shown in Figure 1 through Figure 3.

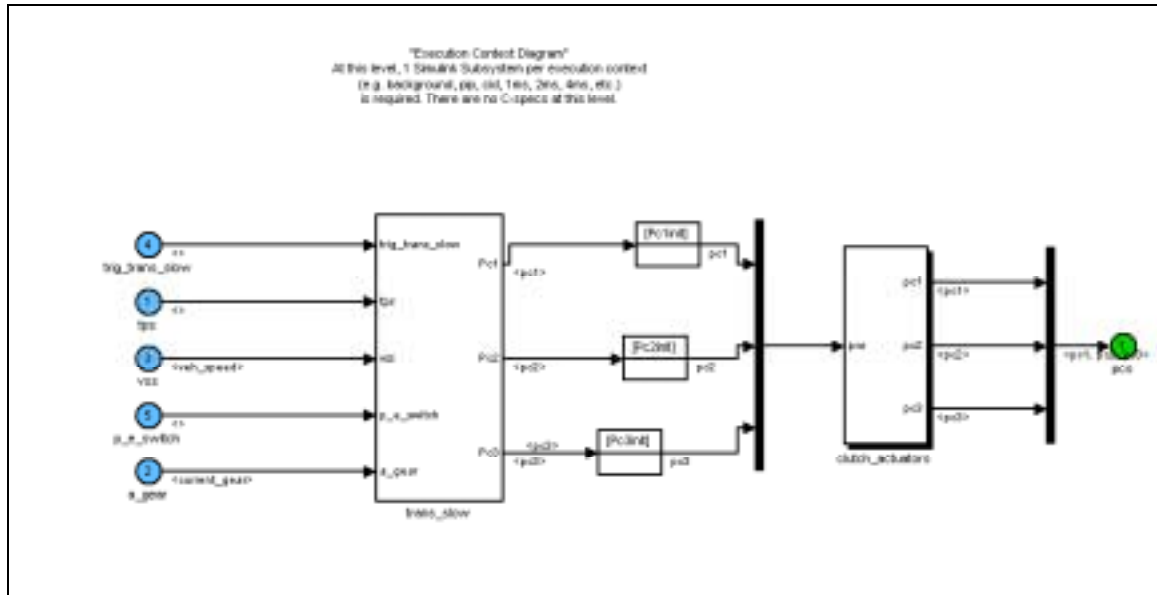


Figure 1: Transmission Controller Model in Simulink

In Figure 1, the subsystem `trans_slow` contains the discrete controller dynamics. This subsystem determines the automatic transmission gear based on the throttle position, the vehicle speed, the current transmission gear status and the status of the performance-economy switch. In addition, this subsystem commands the appropriate clutch pressures to ensure that the transmission is in the desired state. The subsystem `clutch_actuators` is a simplified model of the clutch actuator dynamics and consists of continuous-time integrators. The core of the transmission controller consists of discrete shift dynamics and is shown in Figure 2 and Figure 3. As is clear from these figures, the transmission controller consists of discrete event (logical) and continuous dynamics and hence qualifies as a hybrid system. The main task is to translate the individual pieces in this model to appropriate HSIF constructs.

- References
- O. Cho and J.K. Hedrick, "Automotive Powertrain Modeling for Control", ASME Trans. VIII, Dec. 1985, pp. 560-576.
 - Butts, K., "Analysis Needs for Automotive Powertrain Control," Proceedings of The 7th Mechanisms Forum International Conference, September 6-8 2000, Atlanta, Georgia.

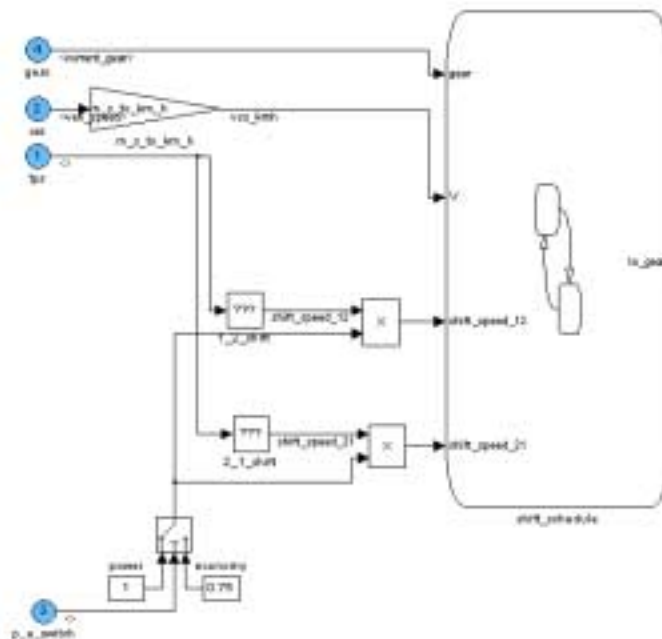


Figure 2: Core Discrete Shift Controller

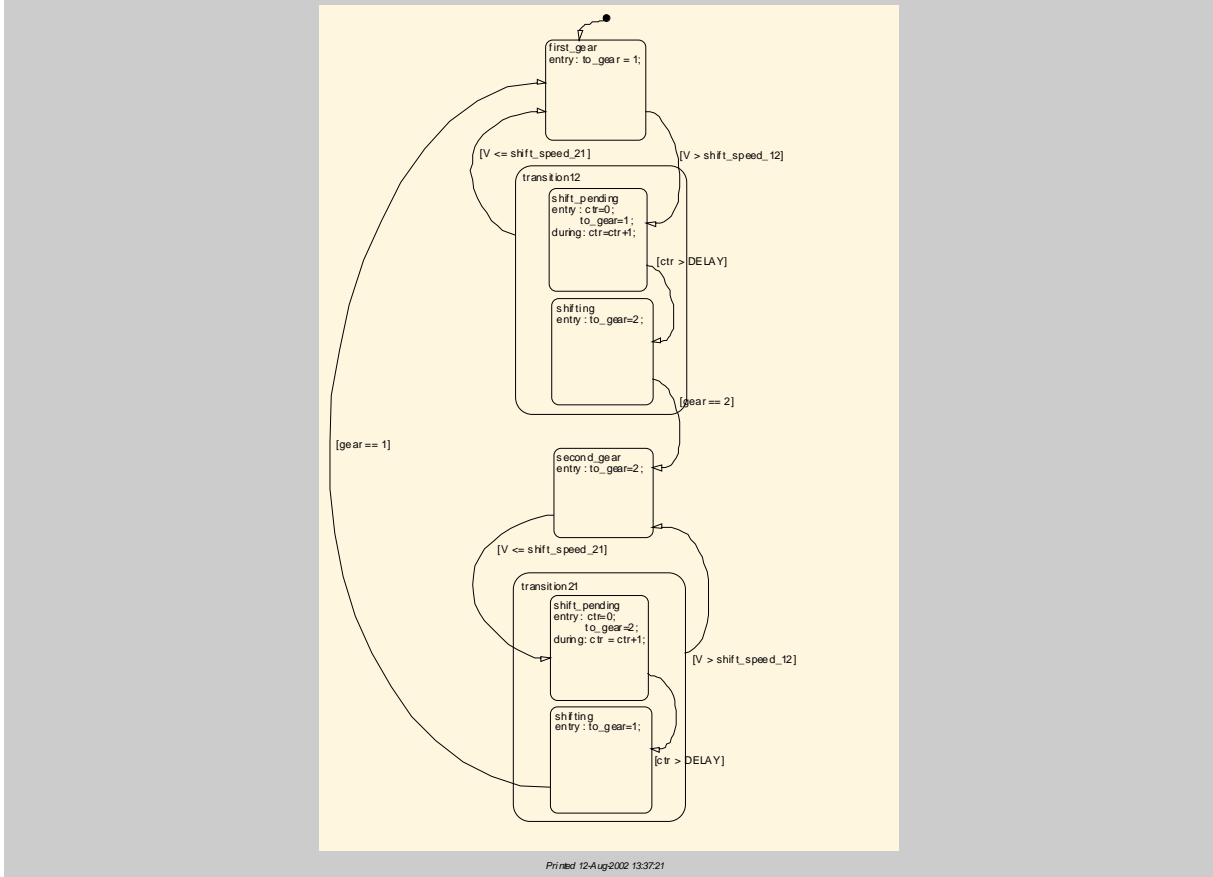


Figure 3: Shift Controller in Stateflow

3. HSIF Model Implementation in GME2000

The HSIF semantics has been used to create a syntax document. This syntax document is in the form of a UML class diagram and defines entities and relationships between the entities [3]. This syntax has been implemented as a Model Paradigm in the Generic Modeling Environment 2000 (GME2000) by researchers at Vanderbilt University. More information on GME2000 and the concept of Model Paradigm can be found in the tool documentation [7]. The process to use this Model Paradigm is also available in the tool documentation. This report assumes that the reader is already familiar with GME2000 and the HSIF Model Paradigm.

4. HSIF Modeling Constructs for the Transmission Controller

There are several constructs in HSIF such as DNHA, HA, Global and Local Parameters, Global and Local Variables, States, Algebraic Equations and Flow Equations. Detailed information on these constructs is available in the HSIF syntax document [3]. The Global variables are variables that need to be used across several Dynamic Network of Hybrid Automata (DNHA). In the benchmark example, the inputs and outputs of the controller obviously fall under this category. In addition, the grade variable (which is used in the plant) is also called out as global since this model can be extended in future to include the complete closed-loop system. The Global variables are throttle pedal position (tps), grade, current gear in the transmission (a_gear), vehicle speed (vss), performance economy switch (p_e_switch), speed multiplier (SpeedMultiplier) and the clutch pressures (Pc1, Pc2 and Pc3). There are no Global parameters. The controller model has two Hybrid Automata (HA). The first one switches the value of the speed multiplier variable based on the performance

economy switch value. The second HA is the core of the transmission shift controller and it decides the gear number based on vehicle speed and throttle position. This HA is shown in Figure 4. Notice that the states in the Stateflow model easily translate to States inside the HAs. Also, note that there are several variables and parameters (such as $Pc_i\tau$, $Pc_{i\max}$ and $Pc_{i\text{state}}$ where 'i' ranges from 1 to 3) which are elements of the same set and which could be concisely represented using vector notation. The parameters DELAY and $m_s_to_km_h$ are local parameters in the sense that these parameters are not required outside the context of Controller HA.

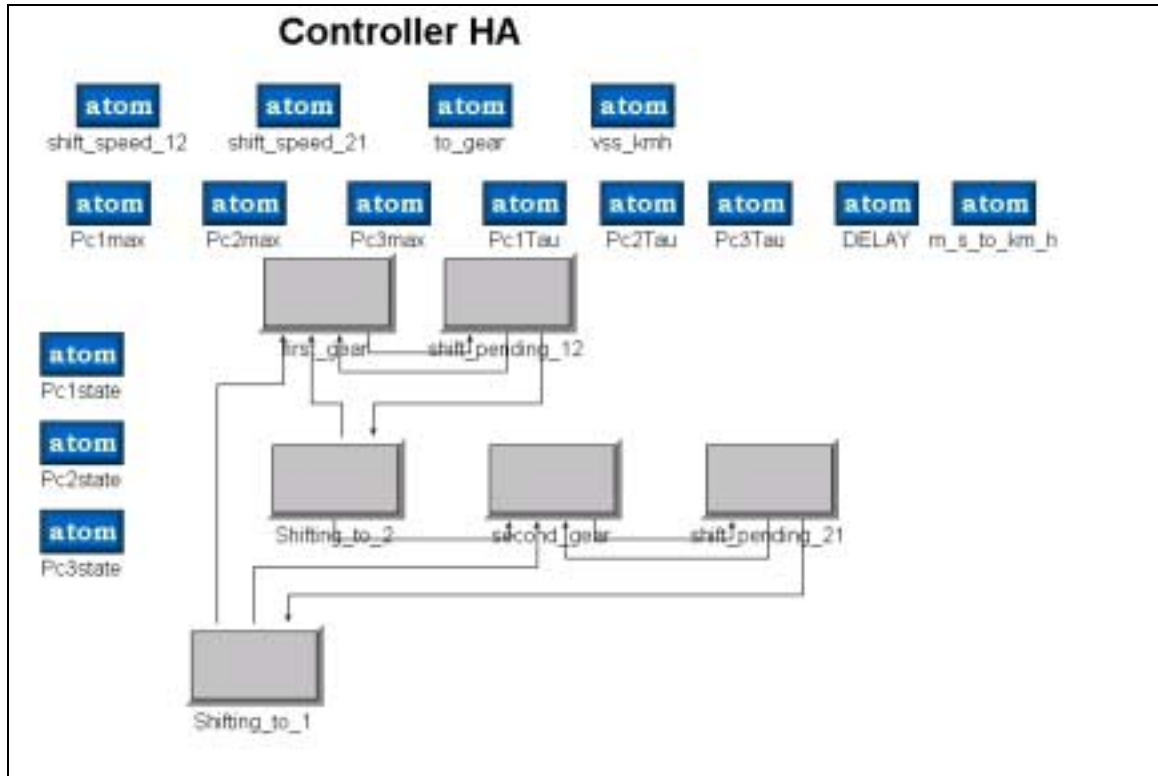


Figure 4: Main Controller HA in the HSIF model

The gray boxes in Figure 4 represent the states of the HA. One such state is shown in Figure 5. This State represents the shift_pending state in the transition12 state in the Stateflow diagram. The algebraic expression for $vss_kmh_compute$ computes the vehicle speed in kilometers-per-hour from meters-per-second and this is a representation of the gain block $m_s_to_km_h$ in Simulink. The algebraic expression for $shift_speed_21_compute$ is currently a placeholder for the $shift_speed_21$ computation involving a look-up table. Since HSIF at this time does not have a syntax defined for a look-up table, this algebraic expression is just a placeholder and is not a valid HSIF element. The $Pc_i\text{StateEquation}$ (where 'i' ranges from 1 to 3) represent the clutch state dynamics and are essentially first-order differential equations with time constant $Pc_i\tau$. Note that the clutch state dynamics are active in all the states of the transmission controller. Hence, the Flow Equations ($Pc_i\text{StateEquations}$) and some Algebraic Equations ($vss_kmh_compute$) are repeated in all the States in the Controller HA. At this time, all of these equations are identical (but repeated) and they use parameters local to Controller HA.

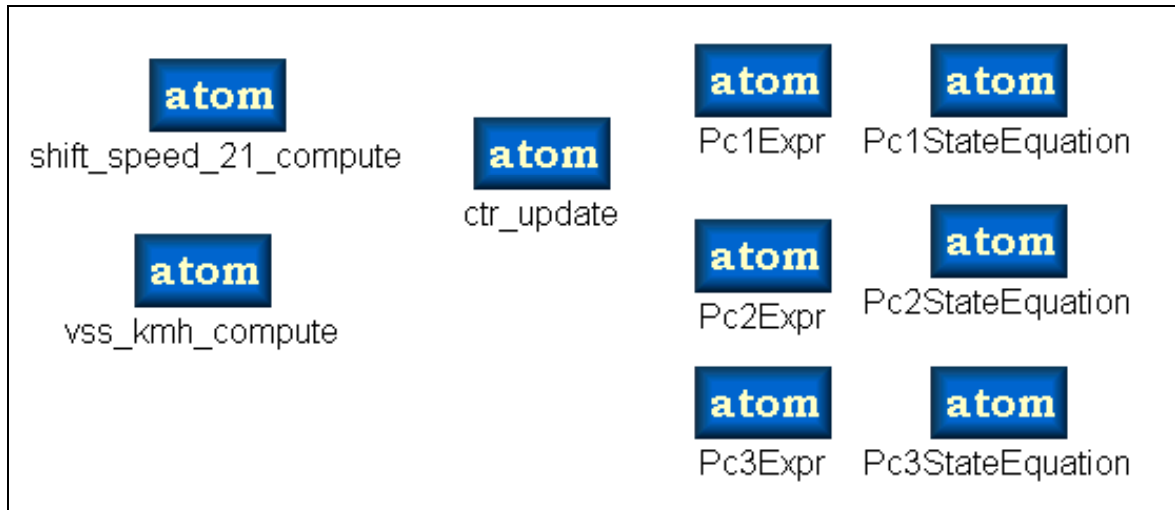


Figure 5: Shift 1 to 2 Pending State

Currently, there are efforts underway to create import and export procedures (and scripts) from individual Hybrid Systems Tools to HSIF representation in GME2000. As part of this exercise, it was decided to explore the HSIF to CheckMate translator. There is no CheckMate to HSIF translator at this time. If an HSIF model exists in GME2000, the HSIF to CheckMate translation process consists of the following two steps:

- 1) Use the Interpreter feature within GME2000 and create an ASCII text file. The HSIF to (Checkmate-compatible) ASCII generation code needs to be installed in GME2000 in order to execute this step. This code has been developed and maintained by Vanderbilt University [8].
- 2) Use the ASCII to Checkmate translator on the file generated from the the previous step and create a CheckMate Simulink model and associated MATLAB files. The MATLAB scripts required for this step has been created and maintained by researchers at Carnegie Mellon University [9].

Before testing the process on the transmission controller HSIF model created during this exercise, it was decided to test some examples included in the tool set. Two such examples were tested. The first one called 'bounce' was originally a CheckMate example that was hand-coded into HSIF by one of the researchers. The second example called 'swimmingpool' was originally a Charon example that was hand-coded into HSIF by one of the researchers. Since the 'bounce' example was hand-coded into HSIF, this exercise would not test the 'round-trip' process of taking a problem from tool X to HSIF back to tool X. In future, when the CheckMate to HSIF translator is available, the 'bounce' example would be a good candidate for such an exercise.

The two examples did not successfully convert into CheckMate in the first attempt. It was found that the ascii file generated from HSIF did not fully conform to CheckMate rules. Also, the ascii to CheckMate converter had some bugs which had to be sorted out. The researchers at Carnegie Mellon University and Vanderbilt University were very helpful in resolving these issues. The ascii file for the 'bounce' example was manually modified by one of the researchers and then the CheckMate model was generated successfully using the second step. The 'swimmingpool' example is currently being analyzed by both the research groups to understand the reasons for the failure of the translation process. Since these are just preliminary versions of these developmental tools, it is expected that future versions would be more robust and user-friendly.

5. Discussion

Based on our experience with the current HSIF semantics and building a model in GME2000, we separate the discussion in this section to (1) observations related to HSIF semantics and syntax and (2) observations related to tools that would generate and use models in HSIF.

5.1. HSIF Semantics and Syntax

The current semantics is a good start and includes the viewpoints of both software and control systems researchers. However, the semantics needs to evolve to handle many realistic models and problems that are of interest to industry.

- In order to share models across groups, there has to be a notion of data that are available to the outside 'world' and data that are expected from the outside 'world'. In many physical modeling tools such as Dymola and AMESIM, the concept of port is used. In control systems modeling tools such as Simulink/Stateflow, StateMate and SystemBuild, the concepts of input and output are used. Currently, the Global variables have an attribute called 'kind' which could be set as 'input', but, there is no notion of 'output'.
- The meaning of 'Input' kind for Local variables is not clear. By definition, Local variables have local scope and if they are inputs, then the meaning of these variables in the parent context is not clear.
- Parameters and Variables need to have another attribute related to dimension. Several modeling and analysis tools can handle multi-dimensional entities such as vectors and matrices. Also, multi-dimensional representation of variables and parameters can lead to compact notation. In the implemented example, it is clear that continuous state variables (clutch pressures), their initial conditions and the corresponding time constants could be concisely represented using vector notation.
- In some real world problems, it is very hard to analytically describe the input-output behavior of a functional element. In such cases, data-based interpolation models (or Look-up Tables) are used to describe the behavior. Based on experimental data, the output variable of interest is described as a function of inputs at discrete input conditions. The model would then interpolate the output if the inputs fall between the discrete grid points. Various interpolation schemes are used to implement such models. In automotive examples, such models are used to describe complex phenomenon such as engine torque generation, torque converter fluid coupling, tire traction characteristics, etc. It would be appropriate for HSIF to define semantics and a corresponding syntax for look-up tables. One of the attributes for a look-up table might be the interpolation algorithm that was used in the 'source' tool.
- HSIF needs to support hierarchy within discrete states. In a number of instances, there are multiple sub-states within a primary state. Although all of these could be considered as primary states, the natural hierarchy leads to a compact representation of the model. Many modeling and analysis tools already support such constructs.
- HSIF needs to support "parallel" states within a HA. In the current HSIF semantics, only one of the states with a HA could be active at a given instant. There are a number of real-world examples where states of unrelated functions can be simultaneously active. The only way to implement such "parallel" states in the current version of HSIF is to have separate HAs for each of the parallel states or create multiple states to represent all feasible combinations of the parallel states. The first approach cannot be used when there are parallel sub-states within a primary state.
- Syntax supporting Local Parameters and Variables inside discrete states is needed. The flow and algebraic equations used inside discrete states may sometimes be repeated across several states. However, in each of those representations, the form of the Flow and Algebraic equations remain the same while the specific parameters may change. Currently, the only way to implement this would be to have individual copies of equations in all states. Any change in the structure of the equation requires changing every instantiation. Instead, if local parameters are allowed, one could have multiple instances of the same equations and then locally change parameters on an as needed basis. This would help in the maintenance of the HSIF model.
- Syntax supporting modeling of discrete system dynamics is required in HSIF. The current Flow Equation syntax allows for differential equations. Some of the models may require difference equations to be modeled. However, the current HSIF syntax does not support this.

5.2. Tool Related

Although the HSIF was conceptualized as a mechanism to exchange information between the three MoBIES tools, we feel that it has potential to serve as an interchange format across several tools (both commercially available and research tools). In order to be an interchange format for a broader set of tools, HSIF should allow for representations that go beyond what the three tools can currently support in their analysis. The

individual tools that import and export to HSIF can selectively approximate (or ignore) those constructs that are not currently supported and warn the user regarding the same.

A lot of existing modeling tools such as Stateflow use built-in conventions to schedule the execution of different parts of the model. Although such scheduling procedures are not widely published, they are available from the developers. It is well known that the behavior of discrete-event systems is strongly influenced by the scheduling algorithms. So, some kind of annotation feature is required in the HSIF syntax that allows for comments regarding any built-in simulation conventions that were used in the 'source' tool. If the destination tool can use such information to recreate the model and force certain scheduling (of the simulation), then the annotation feature might be vital to match the behaviors in two different tools. This annotation feature could be an optional feature that can be populated by the model import/export script.

The success of HSIF would be measured by its adoption and usage by tool vendors and the industry. HSIF would be used for information transfer between tools. The end-user would not directly interact with HSIF but would work with the results of HSIF translations. Hence, there needs to be robust, up-to-date translators for the final implementation. In order to test the processes, it is important to have good preliminary translators. The results so far of using the translation process has been mixed. The tool researchers should work together to complete the preliminary versions of the 'round-trip' translators soon and then put in place a process to update these translators when the tools get updated.

6. Conclusions

The development of an information exchange format for Hybrid Systems models is on the right track with HSIF. This format has a lot of potential to become some sort of a standard. It is necessary to add appropriate content and features to the existing HSIF semantics and syntax to make it useful for all the Hybrid Systems modeling and analysis packages. Some recommendations are made in this report to move HSIF in this direction.

7. Bibliography

- [1] A. Chutinan and K. Butts, "Dynamic Analysis of Hybrid System Models for Design Validation," Smart Vehicle Baseline Report, April 30, 2002.
- [2] University of Pennsylvania MoBIES team, "HSIF Semantics," Version 1, 2002.
- [3] G. Karsai, "Hybrid System Interchange Format (HSIF)," HSIF Syntax document published as part of HSIF v. 1.2 Release, June 2002.
- [4] University of Pennsylvania MoBIES team, "HSIF Semantics," Version 3, Synchronous Edition, August 22, 2002.
- [5] MoBIES group web site at UC Berkeley as viewed on August 26, 2002, <http://vehicle.me.berkeley.edu/mobies/>.
- [6] MoBIES Automotive Powertrain Open Experimental Platform web page as viewed on August 26, 2002, <http://vehicle.me.berkeley.edu/mobies/powertrain/>.
- [7] GME2000 web site as viewed on August 26, 2002, <http://www.isis.vanderbilt.edu/Projects/gme/default.html>.
- [8] HSIF2CM 1.1.1 release, on MoBIES website at Vanderbilt University as viewed on October 1, 2002, <http://www.isis.vanderbilt.edu/Projects/mobies/filedownloads.asp>.
- [9] Ascii to CheckMate download website at Carnegie Mellon University as viewed on October 1, 2002, <http://www.ece.cmu.edu/~webk/cmdev/zhi/ascii2checkmate2.zip>.