A methodology for Integrating and Synchronizing the System Dynamics and Discrete Event Simulation Paradigms

Magdy Helal*, Luis Rabelo*, Jose Sepúlveda*, Albert Jones**

*Department of Industrial Engineering & Management Systems,
University of Central Florida, Orlando, FL, 32816
mhelal@mail.ucf.edu , lrabelo@mail.ucf.edu , Sepulved@mail.ucf.edu
**National Institute of Standards and Technology
Gaithersburg, MD, 20899
albert.jones@nist.gov

Abstract: *With the adoption of integration and system perspectives in managing the manufacturing systems and the pressure imposed by the increased competition and rapidly changing business environment, the need has arisen for new approaches for simulating the manufacturing enterprise. We have proposed SDDES; a hybrid System Dynamics Discrete Event Simulation approach to simulating the integrated manufacturing enterprise. SDDES offers comprehensive simulation models that encompass all management levels and recognize the differences between them in terms of scope and frequency of decision making as well as the levels of details preferred and used at each level. SDDES maintains the integrity of the two simulation paradigms and can use existing/legacy simulation models without requiring learning new simulation skills. In this paper we describe the modular structure of SDDES, our method to synchronize and coordinate SD and DES, and the functional model of the SDDES controller, which manages the integration of the two simulation methodologies.*

Keywords: Hybrid continuous-discrete simulation – Manufacturing enterprise - System dynamics – Discrete event simulation

1. Introduction

The advances in information and computing technologies in addition to the unprecedented levels of competition and fast pace of changes in the business environment have created a more flattened enterprise system and changed the way enterprises should be managed. This is creating challenges to using simulation tools. The presence of manufacturing and non-manufacturing functions in the manufacturing enterprise, the different types of behavior in the system, differences between management levels in the scope of planning, frequency of decision making, and levels of details they deal with, indicate that a single simulation approach can not offer all that is needed in a simulation of such a complex system.

The traditional use of discrete event simulation (DES) to simulate the manufacturing systems has narrowed the scope to detailed statistical analyses at the operational levels of the system. At that level, the main concerns have been the flow of materials and the levels of inventories and work in process, and the performance at the individual level of resources, units of products or processes (Smith, 2003; Bonder and McGinnis, 2002; Lee et al., 2002a; Wu, 2002; 1992; Law and Kelton, 2000; Kosturiak and Gregor, 1999; De Souza et al., 1996; Pegden et al., 1990; Carrie, 1988). The need to simulate the whole system (aggregate and detailed management level functions) has challenged DES, which was inadequate to approximate the continuous behavior in the system generally and particularly at the aggregate levels, or communicate appropriately the financial computations to higher management (Lee et al., 2002a; Johnson and Eberlein, 2002; Barton et al., 2001). DES works effectively with problems of narrow scopes, but it is "… incompatible with a global point of view" (Lin et al., 1998). And it does not address the stability of the system (Rabelo et al., 2005) that should be considered at an aggregate level before any detailed analyses can be conducted (Towill and Edghill, 1989). Add to that its high demand for data and its tedious data preparation processes. There are always detailed data available for the manufacturing functions. But for the aggregate management levels data is not usually available and in most cases only rough estimates exist (Zulch *et al.,* 2002; Mandal and Sohal, 1998; Anthony et al., 1989).

Meanwhile, SD has been successful as a system thinking approach that targets top management levels with a comprehensive integrative perspective, with relatively minimal data requirements. SD is appropriate for modeling large scale systems and the higher levels of decision making where aggregation is preferred. It focuses on the policy decisions that are embodied in the feedback loops, and not on individual localized decisions. Nevertheless, using SD at the operational level of the manufacturing system has failed to offer the needed granularity (Godding et al., 2003; Barton et al., 2001; Baines and Harisson, 1999; Bauer et al., 1982). The same was observed by Choi et al. (2006) who could not use SD to model the performance of the individual processes in a software development system. In addition, while SD permits the study of the stability of the system over the long range, the trends of behavior that it generates do not indicate what specific actions to be made and at what values of the action parameters. Such specifity requires more detailed considerations that SD does not seem to work with, while DES has been effective at.

Consequently, we have argued (Rabelo et al., 2005; Helal and Rabelo, 2004) that SD and DES can complement each other to offer the needed tools to meet the needs created by the modern, integrated manufacturing enterprise system. In this paper, we describe the details of our proposed SDDES simulation method.

The rest of the paper is organized as follows: Section 2 provides a review of literatures supporting the proposed methodology. Section 3 describes the modular structure of the SDDES simulation method and the formalism to describe and communicate the SD and DES modules. Section 4 proposes a new synchronization mechanism to coordinate SD and DES. The synchronization mechanism explicitly recognizes and respects the integrity of the two simulation paradigms. In Section 5 the SDDES controller is described. The controller administers the integration of the simulation modules and offers the user interface. Section 6 concludes this paper.

2. Literature Review

Simulating manufacturing systems has been dominated by DES. Many contributions SD can make in this area, especially integrated systems. SD can capture the causal relationships that are not captured by other approaches that are based on the flow diagramming approaches; DES and other object oriented techniques (An and Jehn, 2005). Further SD offers the ability to model qualitative and *soft factors* (e.g. level of commitment, level of management support, etc.) as was showed by Sterman et al. (1997) who modeled the management's attitude towards various departments in the company after implementing a Total Quality Management program.

Data related to manufacturing activities in manufacturing systems are always available and at very detailed levels where as data for the non-manufacturing functions, usually at the higher levels of decision making are only available as rough estimates and expert guesses (Zulch et al., 2002; Mandal and Sohal, 1998; Anthony et al., 1989). The strategic level is the least systematic process. Operational level decision making uses current, detailed, accurate data. The tactical level falls in between. This obviously has to do with the expected quality of the simulation results that can be obtained if a data demanding technique such as DES is used at the higher levels of management. SD also facilitates conducting designed experiments at the business level (Ashayeri et al., 1998); something that has been available at the detailed levels using DES.

However, while SD has an advantage at the aggregate levels, it can face problems at the operational detailed levels. SD is not believed by many to be suitable for the more detailed operational level activities (Lin et al., 1998; Wiendahl and Breithaupt 1998). Wiendahl and Breithaupt (1998) experimented with simulating manufacturing systems using techniques based on direct use of the automatic control theory and concluded that such continuous simulation approaches (of the control theory) do not lend themselves easily to model the discrete nature of the manufacturing functions. Lin et al. (1998) attempted to develop SD-based manufacturing system modeling library. For instance, a module in the library is the converting activity module, which has inputs (materials), outputs (products) and time (labor). It is used as a process in a manufacturing line. Inside the module, a stock variable accumulates time allocated to the process while a flow variable represents the passage of the time. Further, this module is a part of a bigger system for which a global time is observed and the passage of time is executed at different levels. This was applied to a job shop in a test of the addition of new equipment; an application where DES has been effectively applied for decades. The manufacturing system was not viewed as a whole; to include the aggregate level. Thus SD did not contribute much. In fact this negatively affects SD's intuition, introduces too much details not consistent with SD, and it is an approximation of DES capabilities while DES already exists.

Keenan and Paich (2004) used SD to build a model of GM and the North American auto industry, to assist GM senior management assess the existing policies and improvement initiatives that had been implemented. The initiatives were considered successful but concerns were that the combined impact of them had not met the performance objectives in terms of market share and profitability. The model was comprehensive; including macro-economic variables, market, dealerships, customer behavior, and processes at the various manufacturing facilities located at geographically distant locations. Yet SD as a methodology failed to offer desirable levels of details. Particularly the model could not include the customers' behavior at the household level, which is an important aspect of analyzing the auto market given the high competition and the many brands and models offered every year. The model also could not reach the details of the manufacturing processes and the operations of the dealerships. Only overall measures and indicators were collected regarding the potential policy alternatives. The question that was raised by the senior managers was about where exactly should they intervene and how specifically should the resources be allocated. The trends and the macro level indicators needed to be extended to become actionable.

This has been the case with Godding et al. (2003) who found that the modeling and numerical simulation methods of SD did not provide the needed level of granularity to model the complex stochastic material flows and associated control algorithms for a semiconductor supply network without significant extension. They chose to use DES instead of SD. Bauer et al. (1982)

also found SD limiting their abilities to model the processes at a semiconductor manufacturer and they limited their analysis to localized units. In a different situation, Martin (2001) found that SD could model the environment of the software development process, but for modeling the process itself DES was more effective. He commented: "… we can not ask a SD model questions about the size or complexity of a module of code, because code modules are modeled as individual entities … we can not ask a DES model about the behavior of continuous variables and feedback loops".

On the other hand the challenges facing DES in modeling integrated manufacturing systems are serious. Tow main issues face enterprises as they try to exploit the capabilities of simulation, as observed by the vendors of Arena software. The first is broadening the use of simulation effectively throughout the enterprise in a consistent coordinated way. The second is enhancing the value of the simulation initiatives to the enterprise by leveraging investments in tools and methodologies (Babat and Sturrock, 2003). Gregoriades and Karakostas (2003) and Chang and Makatsoris (2001) suggested limiting the use of DES to certain problem areas; where there are few alternative, short range horizons, and where detailed analysis is needed. Huang et al. (2003) did not believe a single DES model could be used for all of the three levels of management. Lee et al. (2002a) recommended using analytical models for the operational level activities, DES for the tactical level activities, while for the strategic levels they recommended hybrid discrete/continuous simulation models.

2.1 Hybrid Simulation

The limitations of DES at the aggregate levels and in approximating continuous behavior and the limitations of SD at the detailed levels imply the need to develop hybrid continuous-discrete models. Approximating continuous behavior by discrete models can not guarantee accuracy; overestimates or underestimates will likely be obtained (Lee et al., 2002a). The proposed SDDES method is hybrid continuous-discrete approach. There are two approaches to develop hybrid simulations: the hybrid state transition machine (Maler et al., 1992; Harel, 1987) and the DEVS&DES formalism (Ziegler et al., 2000). Both approaches recognize two ways of interactions between discrete and continuous components in a hybrid simulation (See Figure 1).

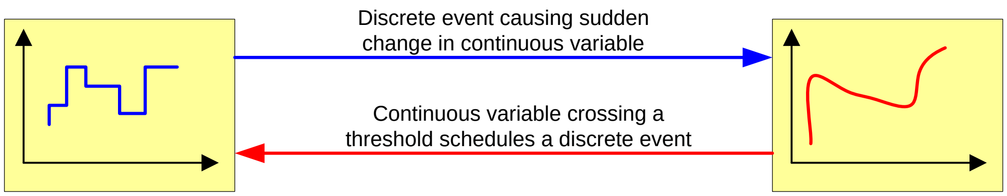

Figure 1: Types of interactions between continuous and discrete components

The hybrid state machine is based on the state chart diagram of the Unified Modeling Language (UML). It is a directed diagram representing states of the system and the transitions between them. Most of its dynamic characteristics were described by Harel (1987). Maler et al. (1992) incorporated the continuous behavior into the discrete state diagram. Some system variables are to be modeled as continuous by differential/integral equations and then threshold values for these variables are defined. During the simulation run when a variable reaches a threshold a discrete *event* is triggered at the state diagram and a transition from a state to a state to another may be taken. Only events can update the system state. Also, upon the execution of an event in the discrete part, a continuous variable can be assigned a new value regardless of its mathematical formulation.

The DEV&DESS formalism combines Ziegler (1976)'s Discrete Event System specification (DEVS) formalism and Differential Equations System Specification (DESS) formalism, to describe hybrid systems. DEV&DESS combines the sets of inputs, outputs, states, and the transition, output, and rate of change functions of the two original formalisms into a unified format to specify the hybrid system. Additionally the DEV&DESS uses the *condition function* to connect the continuous variables to the discrete components of the system. Once a threshold is crossed, the condition function is activated to cause an event to be scheduled at the discrete part and an update of system state to be executed.

The two approaches are fundamentally similar:

1. Both were developed based for control situations; a digital (discrete) system controlling a continuous environment.
2. Running a hybrid simulation is a process of alternating between a discrete phase and a continuous phase. In the discrete phase the state can change but time cannot advance. In the continuous phase the time advances but system state does not change.
3. Events drive the simulation model and only events update the system state.
4. Continuous calculations are performed in the continuous phase between the discrete events, starting with the new state after the event occurrence.
5. The impact of the continuous calculation is communicated to the discrete components by generating a special type of events (state event) based on the values of the continuous variables as compared to predefined threshold values
6. Both favor the use of small-sized objects for which states can be easily enumerated as well as transitions between them.

The behavior of a continuous variable in a hybrid system based on either of the two approaches can be as in Figure 2, in which a continuous variable behaves as continues between events. The (x, y) indicates an event number and its time stamp. The continuous variables must accept abrupt changes in their values by the occurrence of the discrete events. A segment of continuous calculations between two events is not a continuation of the previous segment. Whenever the continuous variable reaches a threshold level, a state event is triggered at the discrete part. Such behavior can be valid in applications such as controlling temperature in an industrial furnace, or controlling the movement of a robot arm and the like, where the threshold approach is applicable. But it is not realistic to expect all continuous systems to be increasing or decreasing until crossing a threshold value. In fact the oscillating behavior of continuous parameters such as inventory, productivity, quality, etc. is more expensive and dangerous in a manufacturing system than a trend in any of them. A threshold can not be used to control such a behavior. Management would work to smooth out oscillations and achieve stability not to prevent a certain critical value if a critical value could be defined. Policies have to be changed and then time should be allowed before realizing the impact of changes. According to SD, stocks can only be influenced by flows over some delay times.

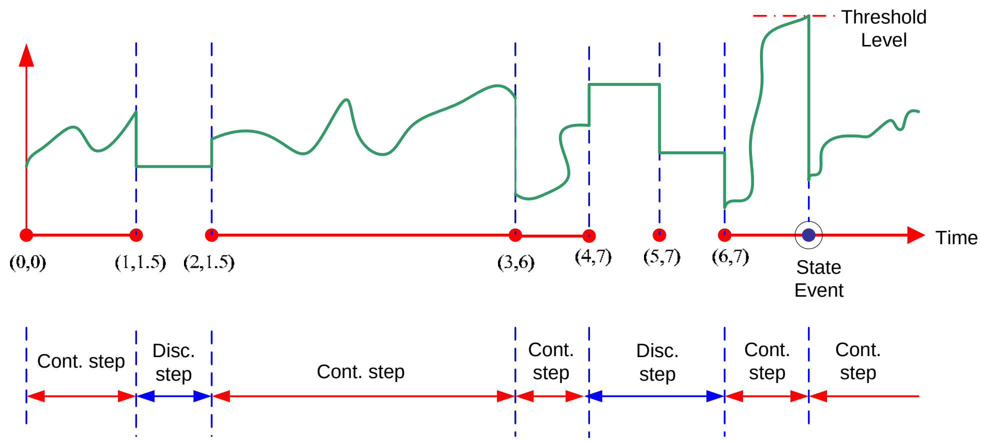

Figure 2: Intermittent continuous behavior in control-based hybrid simulation

2.2 Distributed Simulation

In distributed simulation, loosely coupled simulations interact intensively at certain points in time. Distributed simulation, and parallel simulation as well, are concerned with issues introduced by distributing the execution of DES programs over multiple processors and computing platforms. The technologies were motivated by the needs of the military applications to integrate the geographically distributed systems (simulations and others). Currently the High Level Architecture (HLA) dominates the field as the framework for developing distributed simulations. Still the military uses are main drivers (Bodoh and Wieland, 2003; Borshchev *et al.*, 2002; Fujimoto, 2001; 2000).

In industrial applications, distributed simulation usage is very limited (Boer et al., 2006a; Lendermann, 2006). A recent survey (Boer et al., 2006a; 2006b) has showed that industry practitioners depend heavily on the *commercial-of-the-shelf* (COTS) simulation packages, which offer very limited support for the HLA standard. COTS vendors do not see direct benefits in offering HLA support in their packages, given that HLA is still military-directed and too complex for industry applications. Unlike military applications, industry wants fast and direct results at the lowest expenses, for which HLA as well as the concepts of distributed simulation would add too high overhead technically and financially. Vendors also do not see much economical benefits in collaborating with each other.

Distributed simulation is related to SDDES as SDDES combines two different simulation paradigms. This makes it possible for some causality violations to occur during the simulation run. Distributed simulation uses the conservative (Chandy and Misra, 1978) or the optimistic (Jefferson, 1985) mechanisms for synchronization. Both use events and their time stamps to synchronize the participating simulations. The time bucket (TB) approach was introduced by Stienman (1991) as a simple synchronization approach for CIM settings. The TB allows simulations to advance time in fixed time steps (time buckets) and interact at the end of each bucket. This was inspired by the MRP system where a long-range plan is divided in execution periods/buckets.

Although conservative and optimistic mechanisms as well as the TB method use events, the TB method offered more flexibility when we considered synchronizing SD and DES. TB is consistent with SD as a time driven approach and it is not inconsistent with DES as an event driven approach. Several variations of the TB method have been developed. Stienman (1992) developed a variable size TB method that used events and their consequences to decide the TB size during the simulation run. Fujii at al. (1999) proposed the phased TB method, which used a fixed size bucket but allowed simulations to advance in phases such that a set of simulations (e.g. processes) advance time to the end of the TB then a central simulation (e.g. transporter or a robot system) advance its time while handling data generated by the first set of simulation. Ma et al. (2001) used continuous simulation tools and allowed interactions during the TB. However he limited such interactions to the on/off timeless type of actions taken by a programmable logic controller unit. Bochhima et al. (2005) synchronized continuous and discrete simulations following the threshold approach of Ziegler (2000). TB in their approach was the time between events in the discrete simulations. The continuous simulations run between the discrete events or until crossing a threshold.

In this work we propose a new synchronization mechanism that uses TB concepts. The new mechanism does not need to use events and does not require one simulation paradigm to dominate the other. This is described later in this paper.

3 The SDDES simulation approach

Based on the review of literatures, we argue that:

1. Integrated manufacturing enterprise systems pose challenges to the available simulation tools.
2. DES suffers several shortfalls in offering holistic models of the integrated enterprise.
3. SD assumes aggregate management level perspectives in a systemic integrative approach yet it falls short in adequately modeling the detailed, short-term decision making level.
4. Hybrid continuous-discrete simulation approaches offers the ability to accommodate all types of behavior in the integrated system but they assume control situations and tend to suppress the continuous behavior.
5. Distributed simulation approaches offer favorable features that could improve and facilitate the building of large-scale simulation models of the integrated manufacturing enterprises, but are not yet exploited in such areas for technical and economical reasons.

A combination of SD and DES simulation paradigms has the potentials of satisfying the needed characteristics in the simulation model of the integrated manufacturing enterprise. The integration of SD and DES as proposed in this work offers an inexpensive technique that maintains the existing simulation expertise in simulating the manufacturing systems. Legacy system models can be utilized and no new programming skills are needed to use SDDES. SDDES also does not assume certain situations as do the existing hybrid simulation approaches.

The following sections describe the components of the SDDES method. Figure 3 depicts a roadmap in which the review of current practices in using simulation approaches has indicated the potentials of integrating SD and DES. SDDES is a hybrid continuous-discrete method for simulating the manufacturing enterprise. The size of the simulation model suggested following a modular structure. Modules should be formally described for better model management and communication among modelers. The SDDES formalism is proposed for that purpose. And since SDDES combines two different simulation paradigms, it should also be viewed from the perspective of a distributed simulation arrangement. Specifically, the SDDES synchronization mechanism is proposed to coordinate and synchronize the interactions among the SD and DES simulation modules. The SDDES controller is the main unit in the SDDES simulation method. It manages the integration, implements the synchronization, and offers the user interface. The following sections describe SDDES as implied by Figure 3.

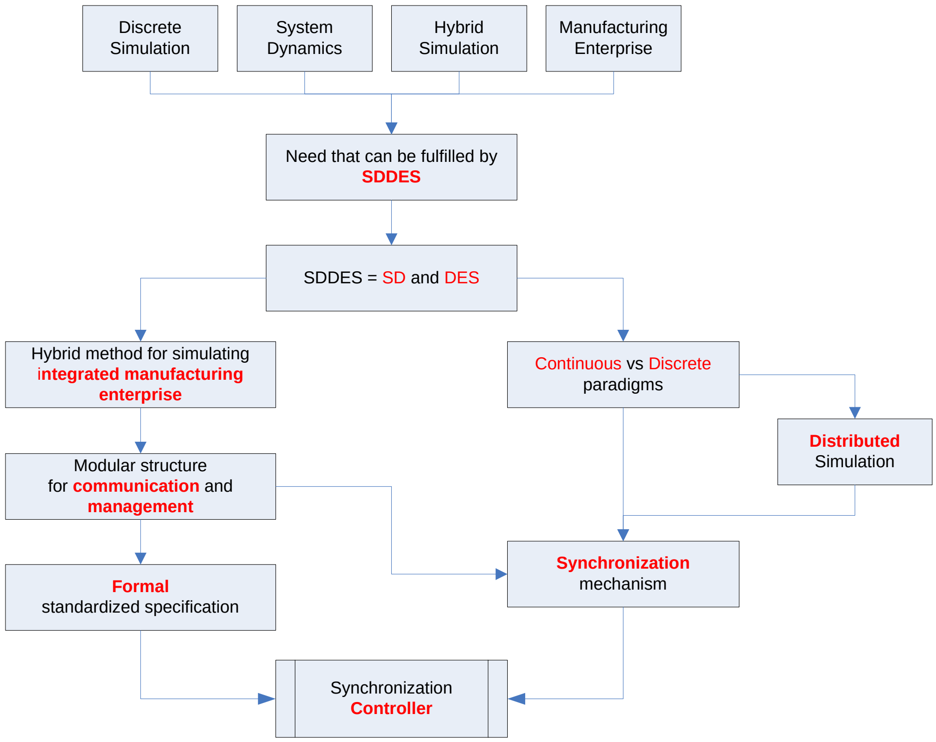

Figure 3: Roadmap toward SDDES

3.1 Layout of the SDDES System

SDDES consists of an overall SD model for the manufacturing enterprise system and a number of DES models built for selected units in the system as dictated by the analysis needs. The models will interact through the SDDES controller. Figure 4 shows the SD model at the top (divided internally into a number of modules; 3 in the Figure), the SDDES controller in the middle, and a number of DES models (3 in the Figure) at the bottom.

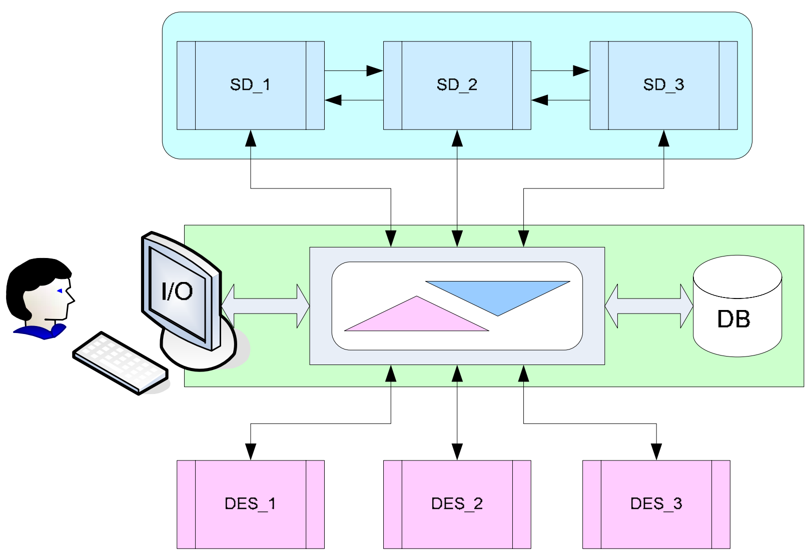

Figure 4: Layout of the SDDES system

Management would use the model for the purposes of testing policies before deciding on implementing them, to confirm the estimated behavior based on enterprise-wide feedback. Management would use the model to investigate the feasibility and desirability of the various programs and initiations in the organization as well as use it to test the resource allocation process to decide on the strategic plans. The feedback structure in the model, in addition, helps better understand the dynamics of the interactions among the system components. In addition, the model can be used as a comprehensive performance measurement system and this can be accomplished by building a balanced scorecard at the top of the model components and parameters. Helal and Rabelo (2004) have discussed the potentials of building dynamic balanced score cards based on SDDES.

3.2 The SDDES Modules

Modules facilitate the model building process, especially when modeling the complex structure of the manufacturing enterprise. They also simplify modifying and extending the model when desirable. Each module has a well defined function, which should have a set of inputs and outputs to be ready to interact with other. The stock management model (Sterman, 1989, 2000) offers a good basis to modularize SD models. DES modules are defined based on the functions they will perform and built from scratch if do not already exist.

3.2.1 The SD Modules

If SD models already exist then they can be modularized such that they are based on the stock management model. New models can utilize the stock management model. We recommend, but it is necessary in SDDES, the use of the stock management model to standardize the building of SD models. Sterman (2000) listed and explained numerous examples for using the stock management model in business systems and other types of systems. Modularizing is an iterative process (Figure 5). The outcome of the process is a set of modules with the inputs and outputs that are to be exchanged between them so they function correctly. The defining of modules should be simultaneous for all SD modules and the DES modules, if necessary.

It should be noted that the modularized SD model is still a single SD model. Modules are logical for the purposes of integration with the DES models and for communication and model management uses. SD modules are treated as sections in the SD model, yet are formalized (using the SDDES formalism) and defined at the SDDES controller as separate units. Not actually dividing the SD model maintains the integrity of the feedback structure of SD while simplifying working with the model in SDDES.

The module represents a function. The inputs to a module are variables that are not normally under the control of the unit manager, or normally included in the core definition of that function represented in the module. To recognize the interactions that unit would have with other units the definition of it could need to be modified based on the requirements of the other modules, following the iterative process of Figure 5.

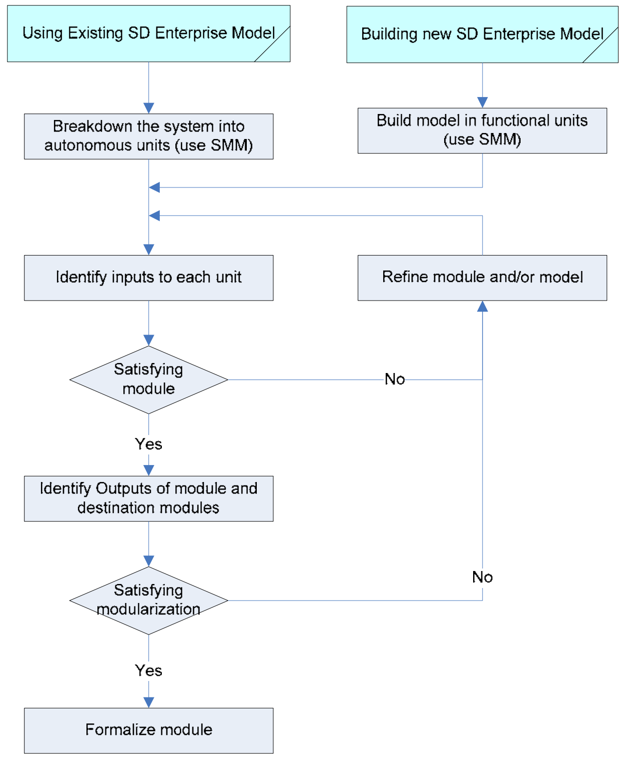

Figure 5: Developing SD modules

The generic SD module can be represented as in Figure 6. Inside the module is a stock management model that is a model of the function of the module. Not all details need to be shown. Yet the sets of inputs and outputs must be well defined. The In_Port_x represents the input ports of communication where the module receives inputs from other SD or DES modules (x represents the number of the port). These are connected to the appropriate variables in the model. The Out_Port_x represents the output ports of communication where the module offers outputs to other modules. Communications through the ports are managed and synchronized by the SDDES controller. Defining the inputs and output is part of the module definition and they are used in the formalism of the module.

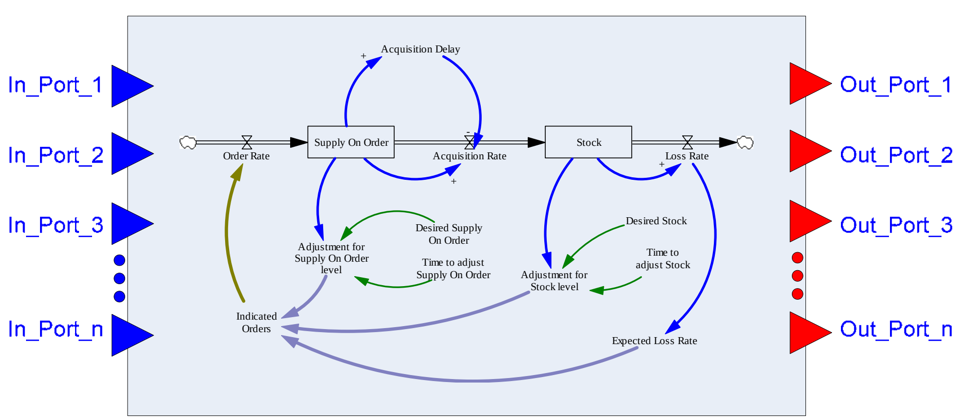

Figure 6: Generic SD module for SDDES

3.2.2 The DES modules

The role of the DES modules in the SDDES is to provide the needed detailed analyses. SDDES allows the use of small-sized DES models instead of the usual practice of building one large DES model of all production functions. This greatly cuts the model development time and expenses. The DES modules are built to interact with one or more SD modules or with each other. The DES modules are complete discrete simulation models that are of narrow scopes. An iterative process to develop DES modules is shown in Figure 7. It starts with a valid DES model for the function of concern. Inputs and outputs and modules to interact with are then identified then the modules are described using the SDDES formalism. Figure 8 shows a representation of a generic DES module. Any DES tool can be used. In this work Arena (http://www.arenasimulation.com/) is used in building DES modules while Vensim (http://vensim.com/) is used for the SD part.

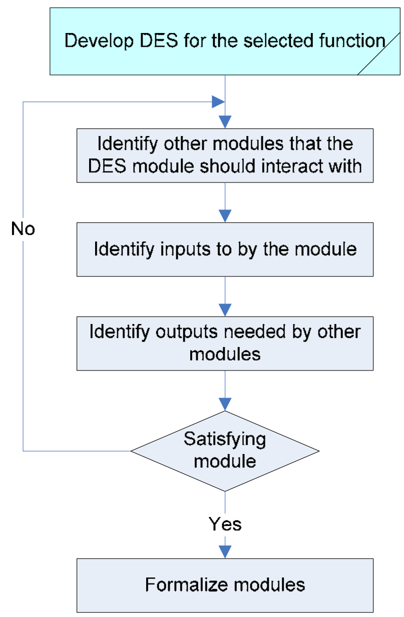

Figure 7: Module development process for DES modules

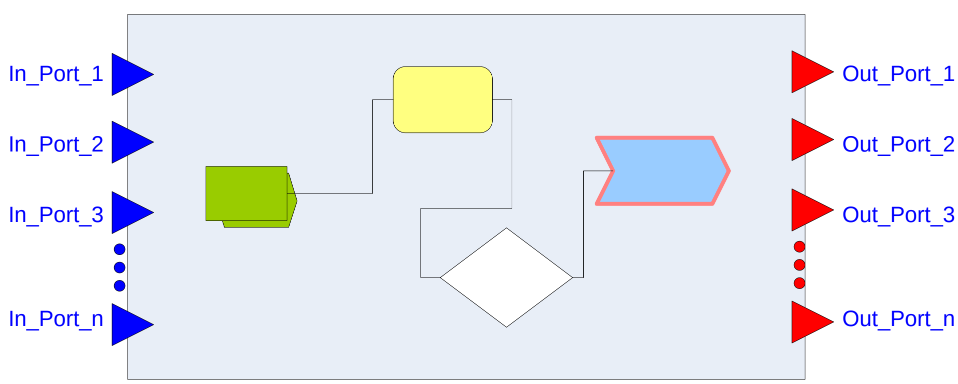

Figure 8: A generic DES module

3.3 The SDDES Formalism

For better model development, management, and communication, the SDDES formalism is proposed to offer a generic description of modules. It offers a standardized structure to build and prepare the interactions of the modules. Users and modelers would only need to provide the specific information to be plugged in the placeholders in the formalism in order to distinguish one module for the others. The formalism is also necessary for the development of a user interface (part of the SDDES controller). The formalism dictates what information the user should provide to define modules. The data of the inputs and outputs and their ports as stated by the formalism are inputted at the controller's user interface along with the type of formatting needed and the timing of the transactions. The functions of the SDDES controller as described later in this chapter depend on the accuracy of defining the formalisms for the modules.

Three sets and two descriptive elements of information are needed in the SDDES modules formalism. Unlike the DEV&DESS formalism, states or state transition functions are not parts of the SDDES formalism. A module of SDDES is represented as in equation 1 in the SDDES formalism terms, where m refers to a SD or DES module.

$$m = (\mathrm{T}, X, Y, P, TB) \qquad (1)$$

The sets of the SDDES formalism are defined as follows:

- T : type of the module; SD or DES module.
- TB : time bucket (described later) of the module; it indicates the run segment length of a DES module. In case of a SD module it is set to $CONTINUOUS$.
- P : set of all variables in the current module. Any variable can be requested by any other module. The same variable can be sent to more than one module.
- X : set of module inputs; variables received from other modules. X is specified as follows:

$$X = \{(m, v, s, op_s, U_m) \mid m \in M, v \in aP, s \in M - \{m\}, op_s \in OutPorts_s, U_m \subset P_m\} \qquad (2)$$

In Equation 2:

- m indicates the current module
- v is the input variable
- P_m is the set of variables of current module
- aP is the set of all variables in all other modules less the current module. For any module j , $aP_j = \bigcup_{i=1, i\neq j}^{M} P_i$
- M is the set of all modules in the SDDES model
- s indicates the source module from which v is obtained.
- op_s is an output port in the source module through which v is obtained
- U_m is the set of variables in the current module that use the input value. It is defined by Equation 3.

$$U_m = \{(ip, u, t, f) \mid ip \in InPorts_m, u \in P_m\} \qquad (3)$$

In Equation 3:

- ip is an input port in current module
- $InPorts_m$ is the set of input ports of current module
- u is the variable in current module that uses the input variable.
- t represents the timing of reading the input variable to be used by u.
- f indicates the data preparation action that the controller has to perform before sending the input data to the requesting module. There are two types of actions: Aggregate and Disaggregate that are specified by the user when defining the modules.

- Y is the set of outputs. It is described by Equation 4.

$$Y = \{(m, op, v, D) \mid m \in M, op \in OutPorts_m, v \in P_m, D \subset M - \{m\}\} \tag{4}$$

In Equation 4:

- op is an output port in the current module
- $OutPorts_m$ is the set of all output ports in the current module
- v is the output variable leaving at the current output port
- D is the set of modules that receive the current output variable from current module m. Several modules can receive an output variable and several variables in each can use the received value. D is given as in Equation (5)

$$D = \{(m_d, V_d) \mid m_d \in M - \{m\}, V_d \subset P_{md}\} \tag{5}$$

In Equation 5:

- m_d is a module that is receiving an output of module m
- V_d is the set of variables in m_d that use the received value
- P_{md} is the set of all variables in the receiving module

Each element in V_d consists of the identification number of the input port through which the received value is received, the name of the using variable, the timing of reading the received value, and the necessary formatting. The whole SDDES model is described as the set of all modules interacting through the controller. The SDDES model is described by Equation 6.

$$SDDES = \{m \mid m \in M\} \tag{6}$$

3.4 The SDDES Synchronization Mechanism

We propose the SDDES synchronization method to synchronize separate SD and DES simulation models. The proposed method makes use of the concepts of the TB synchronization approach. The conservative simulation approaches depend on using a lookahead interval to determine the safe time advancement step. The optimistic approaches use messages of the timestamps of the events to control the advancement of time and perform rollbacks when needed. They both assume discrete simulations, or a system that is dominated by discrete behavior. Continuous simulation does not generate events and does not have states that can be defined practically, if can be defined at all. Synchronization of the continuous simulations with each other or with discrete models can be approached using TB-based approaches. TB approaches are consistent with advancing time in steps in SD and by following events in DES. The optimistic and conservative approaches depend on the use of events that does not make them fit the synchronization function in SDDES.

TB in this work is related to the DES modules and will be used to define the length of the run for each DES module. The choice of the bucket size will remain an important decision that should be made considering the desirable levels of accuracy and efficiency as well as the nature of the system unit being modeled in DES. An iterative approach to deciding on the TB size is described in Figure 9. A short and large bucket sizes compare to each other as in Table 1.

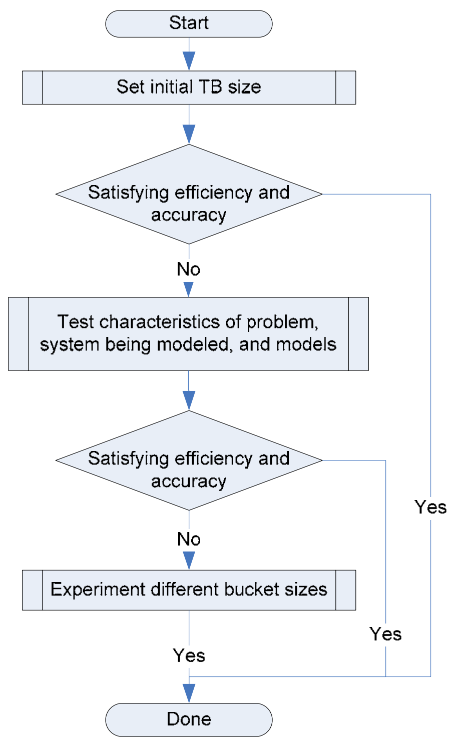

Figure 9: Process of deciding on the TB size

Table 1: Characteristics of large and small bucket sizes

Large Bucket Size	**Small Bucket Size**
Lower accuracy	Higher accuracy
Low fidelity	Higher fidelity
Fast simulation run	Slow simulation run
Optimistic synchronization in nature	Conservative synchronization
Less flexibility in fitting other systems and estimating costs and performance measures (lateness, lead times, rates, etc)	More flexibility in fitting other systems and estimating costs and performance measures (lateness, lead times, rates, etc)

In SDDES we avoid having one simulation paradigm dominating the other. If discrete models are to be dominant then control situations (e.g. discrete/digital control of a continuous process) will be the case. Control situations can not be applicable to all business or social phenomena that are fundamentally continuous and do not normally change abruptly. And in addition the continuous units; mainly at the top levels of decision making, are usually generating the guidelines for the discrete units at the operational or detailed levels.

SDDES is basically a time-driven simulation. This is because the SD unit represents the total enterprise system while the DES units are parts in that system. This is close to real practices where plans set at higher management levels are executed at the lower management levels. The TB is the time advancement step for the SDDES model. It is of fixed size for the SDDES model as a whole but each DES module can have its own TB size. TB is not the same as the time step in the SD technique. The integration calculations in SD are preformed with the appropriate time step while TB is the step for advancing time for the simulation as whole. This allows the accuracy of SD to be set without restrictions from the DES units.

The simulation run length of the SDDES model is defined in the SD part as the planning horizon for the enterprise. Each DES module's run is broken into several run segments. Each segment is a complete discrete simulation run with sufficient number of replications. Each segment is initialized with the status of the DES module at the end of the previous segment in addition to any adjustment received from SD modules at the end of the previous run (TB). Run segments for the same module have the same length but each DES module has a different run segment length (TB). TB of the whole SDDES model is the minimum run segment length among all DES modules. This will be called the base TB for the whole SDDES run and termed L. The run segment length for any DES is nL ,where n is nonzero positive integer. Check points among SD and DES models is at the end of each TB for each DES module. It is also allowable for the modules to send a receive data values during the runs to keep going. These values are planned to be sent or received before the run starts and are needed to maintain logical simulation results. L; the base TB is determined based on operational, managerial, and computational considerations to achieve the best balance between accuracy and efficiency.

Figure 10 describes the SDDES synchronization mechanism, assuming three DES modules interacting with a SD model (which has several modules in it). All interactions in the SDDES model are made through the SDDES controller, which is also indicated in the Figure. The TB sizes for the DES modules are L, $2L$, and $5L$ respectively. The numbers on the arrows in Figure 10 indicate the sequence of actions. The SD part sends the initialization data for all the DES modules at the beginning of the run such that the system starts at steady state. DES modules advance time toward the TB of each. At the end of a TB the modules that have finished a run would send and receive data to each other and to SD; all via the SDDES controller. All formatting is done at the controller side.

All data exchange transactions are defined at the controller with the appropriate user interfaces it provides and with the specification of the SDDES formalism. The controller monitors and records the simulation time for the DES modules as they don't run to the end of the SDDES simulation run, but in separate segments. Functions of the controller are explained in the next section.

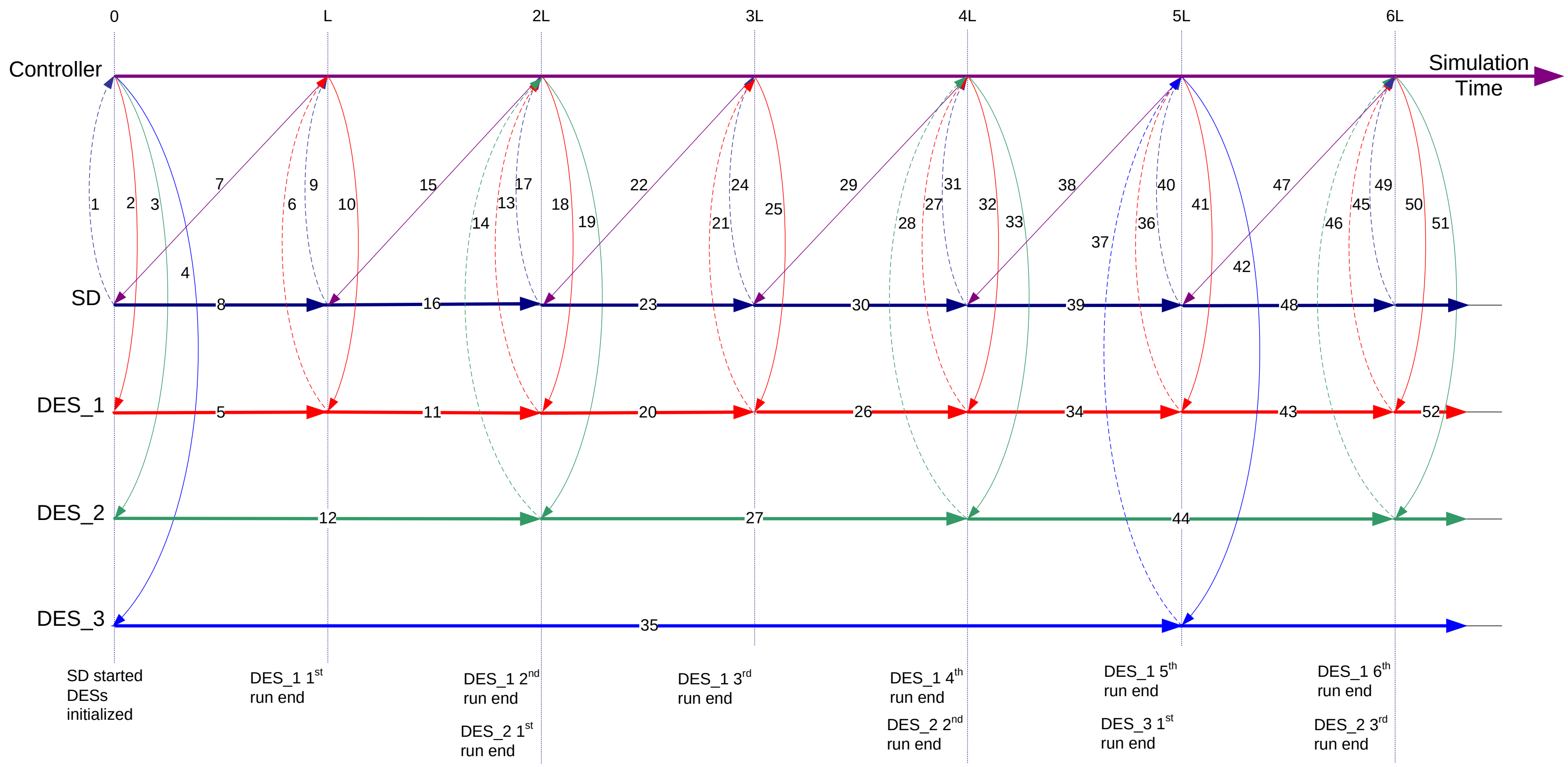

Figure 10: Interactions among SD and the DES models

Interactions among SD and DES as well as among the DES modules are also allowable during the run segments. For instance to release orders and raw materials in DES periodically, SD must send the relevant data at the specified times to DES. This should be done for each replication by saving the SD values to be used in each replication on the right time indicated by SDDES controller. The interaction ports are defined in the controller and this includes the timing of data exchanges.

3.5 The SDDES Controller

SDDES uses the existing SD and DES modeling techniques as they are normally used. The integration of the modules and what it entails are all managed by the SDDES controller. The SDDES controller is the manager of synchronization of the SD and DES simulation modules in the SDDES framework. The controller is a separate unit that interacts with the simulation modules to integrate them and facilitate the interactions between them according to the specifications included in the SDDES formalism. In fact the formalism information is stored in the controller model database. The controller also implements the synchronization mechanism to control the running and stopping of the modules. It is also the user interface to perform I/O operations and to define/modify/replace the modules. Specifically, the SDDES controller acts in the following areas:

1. **Data management**: The controller ensures that the information indicated in the definition of the SD and DES modules (in their formalism specifications) are executed properly, in relation to the formatting of the data.
2. **Time management**: The controller implements the synchronization mechanism to control the running of the modules. It monitors the simulation time. The DES modules do not run for the entire SDDES simulation horizon in a single run, and the simulation times from them are not usable directly. The controller estimates the time for each with respect to the overall SD model such that the user can observe the correct time.
3. **Participation management**: The controller offers the functionality needed to add new modules to the SDDES model as well as to modify or replace already functioning modules. This is achieved through a user interface through which the modeler inputs the necessary data by the SDDES formalism to specify a module.

All functions the controller does are based on the user interactions with its user interfaces. An IDEF0 of the SDDES controller functions and details about how they are performed are presented in the following section.

3.5.1 Functional model of the SDDES controller

The IDEF0 method offers a hierarchal representation of a system that depicts the functions done within the system along with relevant inputs needed to perform the functions, outputs generated upon perfuming the functions, the controls that guide and constrain the functions, and the mechanisms needed in that. The basic model is shown in Figure 11 presented from the point of view of the modeler/the user of the SDDES. The A-0 IDEF0 model is the most abstract representation. A single box is used to indicate the function of the controller, namely Execute SDDES. The SDDES controller fundamentally executes the simulation run that the SDDES model is built to make. The sets of inputs, controls, outputs, and mechanisms (the ICOMs) used in the A-0 model are described in Table 2.

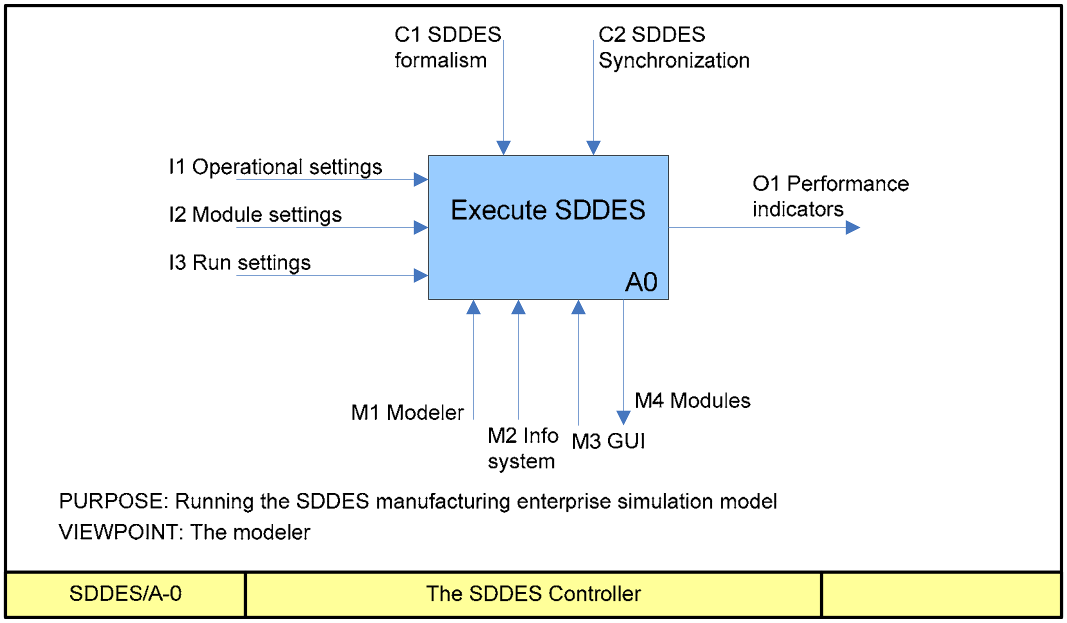

Figure 11: A-0 IDEF0 functional model of the SDDES controller

Table 2: ICOMs for the A-0 IDEF0 model of the SDDES controller

Inputs	I1	Operational settings	Characteristic information representing the current status of the system. They are elements of the management policies that will be tested and evaluated with the simulation model. The module variables are assigned values in this action. These values are provided by the modeler or obtained from the active information system in the company (M2).
	I2	Modules settings	The inputting of the data required by the SDDES formalism. Modules can be modified, deleted, or added to the model.
	I3	Run settings	Specifying the planning horizon, number of replications for the DES modules, as well as the time units and needed parameters that will be monitored. Also the outputs that are of interest are specified here
Controls	C1	SDDES formalism	This guides the addition, modification, or deletion of modules. Also specifies the data needed to set the model and the run.
	C2	SDDES synchronizati on	This is SDDES synchronization algorithm. It guides the simulation run and the data exchange transactions.
Output	O1	Performance indicators	This is the regular outputs of a simulation model
Mechanisms	M1	Modeler	Represents the user of the simulation model in general. The modeler performs all I/O operations

	M2	Info system	This is the existing information system of the company (e.g. ERP or MRP). Module variables are linked to data provided by the information system. Outputs can also be added to the information system.
	M3	GUI	the graphical user interface is an integrated unit of the controller. It offers several user interfaces through which the modeler interacts with the controller and the model.
	M4	Modules	These are the module information saved in their files (e.g. the Arena and Vensim files in the current work). They are called to be used as necessary by the modeler and during the run for sure.

The controller executes the SDDES simulation model using these sets of inputs, controls, outputs, and mechanisms. The modeler uses the appropriate user interface to input module data or the simulation data. The modeler also observes the outputs during the simulation run and can pause the model to modify some settings or stop the simulation for a new experiment. The inputs provided by the modeler are specified by the two controls; the formalism and the synchronization algorithm. The controller contains a database to store the input data and the ongoing outputs during the run. To set the modules, the controller extracts the parameters of the modules (via calling M4) such that the modeler would assign values to them or link their values to the appropriate data in the information system. The decision on the TB for the DES modules can also use inputs from the information system.

The A-0 diagram is decomposed into more detailed definition of the controller functions. The A0 diagram of the IDEF0 model is the first level of details of the function described in the A-0 model. A0 for the SDDES controller models the three basic functions of the controller as described in the previous section; described however in more practical terms. In the IDEF0 terms these functions are the A1, A2, and A3 in Figure 12. Each of these functions is decomposed further as necessary to offer a complete description of the controller role in the SDDES model, prior to its implementation. The A0 model is describe in Figure 12 and explained afterward.

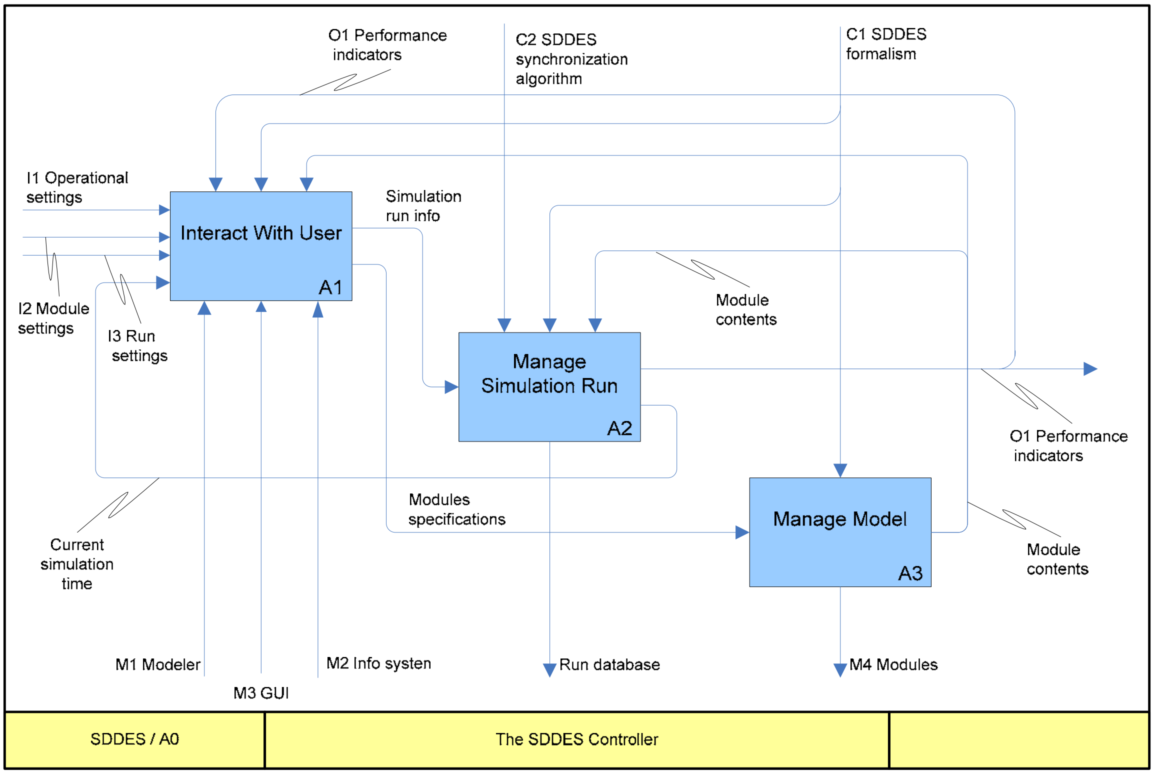

Figure 12 : The A0 IDEF0 model of the SDDES controller

The Interact With User function (A1) allows the user (the Modeler in the above model) to perform I/O operations as well as defining the modules. The appropriate GUI is initiated for the Modeler to input the necessary settings. These inputs are communicated to A2 and A3 for the models to be defined and the run to be ready to be executed. The GUI is developed to meet all use cases of the system and these use cases are controlled by the SDDES formalism (adding or deleting modules), by the current contents of the saved modules (coming from A3 to modify modules, assign input values to their variables, etc.), and by the performance indicators (coming from A2 for the user to observe outputs and do necessary adjustments when desirable).

The Manage Model function works to accept changes in the existing modules and add new ones to the SDDES model as inputted by the modeler in A1. The modules are saved in their simulation software files and the files are called as necessary (M4). The current contents of the modules are the outputs of A3 that are fed back to A2 so that the controller reads the TB setting for the modules and the defined data exchange transactions that will be executed during the run in the A2 function. The synchronization algorithm (C2) controls A2 along with the relevant information from the formalism (C1). The current module contents from A3 are also fed back into A1 for the Modeler to correctly assign the operational and run settings.

The output of A2 is the output of the simulation run, which is offered as the overall output of SDDES and is fed back to A1 for the modeler to analyze the performance with the appropriate GUI. It is noted that the A2 function is internal; no direct user interactions are needed with it. During the run, the behavior of the system is feedback to A1 for the user to perform any adjustment if desirable.

The output of A1 is the simulation run info representing the settings needed to start the simulation run at A2. These ongoing outputs are saved in the run database. They are updated all the time during the run. Of particular importance, the results of the run segments of the DES modules are saved to be used in the following segments.

4.0 Summary and Future Work

We propose a new hybrid simulation methodology that combines the SD and DES simulation paradigm to simulate the manufacturing enterprise system. We described the modular structure of the methodology, the formalism to describe and communicate the modules, the synchronization mechanism, and the controller unit that manages the interactions. The proposed SDDES methodology has the potentials to bridge the simulation gap identified in simulating the integrated manufacturing enterprise system. The proposed methodology can cover all types of behavior in the enterprise system and can accommodate the differences between the management levels in terms of scope and frequency in decision making and the levels of details in data at each level.

The SDDES maintains the integrity of the two simulation paradigms and does not allow one to dominate the other as do the existing hybrid simulation frameworks. SDDES can extend the applicability of SD in the manufacturing system applications as well as enhance the usability of DES in simulating large scale complex systems. Modelers need not learn new simulation skills and existing/legacy SD and DES simulation models can be used in SDDES.

Currently we are working to implement the design described in this paper. Data has been collected from a real manufacturing company, and a comprehensive SD as well as two DES models have been built. The objectives of the experimentation are to validate the usefulness and effectiveness of the proposed synchronization mechanism in particular and the SDDES simulation approach in general.

References

Aguilar-Saven R. 2004. Business process modeling: Review and framework. *International Journal of Production Economics* **90**(2): 129-149.

Alexopoulos C, Kim S. 2002. Output data analysis for simulations. The Winter Simulations Conference, WSC’02, Dec 5-8, Orlando, FL

An L, Jehn J. 2005. On developing system dynamics model for business process simulation. The Winter Simulations Conference, WSC’02, Dec 5-8, Orlando, FL

Anthony R, Dearden J, Bedford N. 1989. *Management Control Systems*. 6th Ed. IRWIN. IL

Ashayeri J, Keij R, Broker A. 1998. Global business process re-engineering: a system dynamics-based approach. *International Journal of Operations and Production Management* **18**(9/10): 817-831

Baines T, Harrison D. 1999. An opportunity for system dynamics in manufacturing system modeling. *Production Planning and Control* **10**(6): 542-552

Barton J, Love D, Taylor G. 2001. Evaluating design implementation strategies using simulation. *International Journal of Production Economics* **72**: 285-299

Bauer C, Whitehouse G, Brooks G. 1982. Computer simulation of production system: Phase I. *Technical Report COE No. 82-83-1*. The University of Central Florida, Orlando, FL

Bodoh D, Wieland F. 2003. Performance experiments with the high level architecture and the total airport and airspace model (TAAM). The 17th Workshop on Parallel and Distributed Simulation, IEEE, June 10-13, San Diego, CA

Boer C, Bruin A, Verbraeck A. 2006a. Distributed Simulation in industry: A survey Part 1 – the COTS vendors. The Winter Simulation Conference; WSC'06, Dec 3-6, Monterey CA

Boer C, Bruin A, Verbraeck A. 2006b. Distributed Simulation in industry: A survey Part 2 – Experts on distributed simulation. The Winter Simulation Conference; WSC'06, Dec 3-6, Monterey CA

Bonder D, McGinnis L. 2002. A structured approach to simulation modeling of manufacturing systems. Proceedings of the 2002 IERC, May 19-21, Orlando FL

Borshchev A, Karpov Y, Kharitonov V. 2002. Distributed simulation of hybrid systems with AnyLogic and HLA. *Future Generation Computer Systems* **18**(6): May 2002

Bouchhima F, Nicolescu G, Aboulhamid E, Abid M. 2005. Discrete-continuous simulation model for accurate validation in component-based heterogeneous SoC design. RSP'05: IEEE International Workshop on Rapid System Prototyping: pp 181-187

Carrie A. 1988. *Simulation of manufacturing systems*. John Wiley & Sons, GB.

Chandy K, Misra J. 1978. Distributed simulation: A case study in the design and verification, of distributed programs. *IEEE Transactions on Software Engineering* **5**(5): 440-452

Chang Y, Makatsoris H. 2001. Supply chain modeling using simulation. *International Journal of Simulation* **2**(1): 24-30

Choi K, Bae D, Kim T. 2006. An approach to a hybrid software process simulation using the DEVS formalism. *Software Process: Improvement and Practice* **11**(4): 373-383

De-Souza R, Huynh R, Chandrashekar M, Thevenard D. 1996. A comparison of modeling paradigms for manufacturing line. IEEE Int. conf on systems, management, and cybernetics, Oct 14-17, Beijing, China

Fujii S, Ogita A, Kidani Y, Kaihara T. 1999. Synchronization mechanisms for integration of distributed manufacturing systems. *Simulation* **72**(3): 187-197

Fujimoto R. 2000. *Parallel and distributed simulation systems*. John Wiley & Sons, Inc. USA

Fujimoto R. 2001. Parallel and distributed simulation systems. The 2001 Winter Simulation Conference; WSC'01

Godding G, Sarjoughian H, Kempf K. 2003. Semiconductor supply network simulation. The Winter Simulation Conference, Dec 7-10, New Orleans, LA

Gregoriades A, Karakostas B. 2004. Unifying business objects and systems dynamics as a paradigm for developing decision support systems. *Decision Support Systems* **37**: 307-311

Haler D. 1987. Statecharts: A visual formalism for complex systems. *Science of Computer Programming* **8:** 231-274

Hannet J. 1999. From the aggregate plan to lot-sizing in multi-level production planning. Brandimarte, P, Villa, A. (Eds.): *Modeling Manufacturing Systems from aggregate planning to real time control*. Springer-Verlag Berlin, Germany

Helal M, Rabelo L. 2004. An enterprise simulation approach to the development of dynamic balanced scorecards. ASEM'04; Proceeding of American Society of Engineering Management Conference, Oct 20-23, Alexandria, Virginia

Huang G, Lau J, Mak K. 2003. The impact of sharing production information on supply chain dynamics: a review of the literature. *International Journal of Production Research* **41**(7): 1483-1517

Jefferson D. 1985. Virtual time. *The ACM Transactions on Programming Languages and Systems* **7**(3): 404-425.

Johnson S, Eberlein B. 2002. Alternative modeling approaches: a case study in the gas and oil industry. The 2002 SD Society Conference, July 27-30, Shanghai, China

Keenan P, Paich M. 2004. Modeling general motors and north American automobile market. 22nd International Conference of the System Dynamics Society, July 25-29, Oxford, England

Kosturiak J, Gregor M. 1999. Simulation in production system life cycle. **Computers in Industry 38**: 159-172

Law A, Kelton W. 2000. *Simulation modeling and analysis*. McGraw Hill, USA

Lee Y, Cho M, Kim S, Kim Y. 2002. Supply chain simulation with discrete-continuous combined modeling. *Computer and Industrial Engineering* **43**: 375-392

Lendermann P. 2006. About the need for distributed simulation technology for the resolution of real-world manufacturing and logistics problems. The Winter Simulation Conference WSC'06, Dec 3-6, Monterey CA

Lin C, Baines T, O'Kane J, Link D. 1998. A generic methodology that aids the application of system dynamics to manufacturing system modeling. International Conference on Simulation, Sep 30 – Oct 2 (IEEE Conf. Pub. No. 457)

Ma Q, Judd R, Lipset R. 2001. Distributed manufacturing simulation environment. Proceedings of the Summer Computer Simulation Conference, July 15-17, Orlando, FL

Maler O, Manna Z, Pnueli A. 1992. From timed to hybrid systems. In Bakker, J, Huizing, C, Roever, W, Rozenberg, G. (Eds.) *Real-Time: Theory in Practice*. Springer-Verlag, Germany

Mandal P, Sohal A. 1998. Modeling helps in understanding policy alternatives: A case. **Journal of Management in Engineering** Jan-Feb: 41-48

Martin R. 2001. *A hybrid model of the software development process*. PhD Thesis, Portland State University

Pegden C, Shannon R, Sadowski R. 1990. *Introduction to simulation using SIMAN*. McGraw-Hill, USA

Rabelo L, Helal M, Jones A, Min H. 2005. Enterprise simulation: A hybrid system approach. *International Journal of Computer Integrated Manufacturing* **18**(6): 498-508

Smith J. 2003. Survey on the use of simulation for manufacturing system design and operation. *Journal of Manufacturing Systems* **22**(2): 157-171

Steinman J. 1991. SPEEDES: Synchronous parallel environment for emulation and discrete event simulation. Proceedings of the SCS Western Multi-conference on Advances in Parallel and Distributed Simulation **23**(1): 95-103.

Steinman J. 1992. SPEEDES: A multiple-synchronization environment for parallel discrete-event simulation. *International Journal in Computer Simulation* **2**: 251-286

Sterman J. 1989. Modeling managerial behavior: misperception of feedback in a dynamic decision making experiment. *Management Science* **35**(3): 321-339

Sterman J. 2000. *Business dynamics: systems thinking and modeling for a complex world*. McGraw Hill, New York, USA

Sterman J, Repenning N, Kofman F. 1997. Unanticipated side effects of successful quality programs: Exploring a paradox of organizational improvement. *Management Science* **43**: 503-521

Towill D, Edghill J. 1989. The use of system dynamics in manufacturing systems engineering. *Transactions of the Institute of Measurement and Control* **11**: 208-216

Wiendahl H, Breithaupt J. 1998. Automatic production control: a new approach in production planning and control based on methods of control theory. Drexl, A, Kimms, A. (Eds.) *Beyond manufacturing resource planning (MRP II).* Springer, Berlin, Germany

Wu B. 1992. *Manufacturing system design and analysis*. CHAPMAN & HALL, GB

Wu B. 2002. *Handbook of manufacturing and supply system design.* Taylor and Francis, London.

Zeigler B. 1976. *Theory of modeling and simulation*. Wiley, NY

Zeigler B, Praehofer P, Kim T. 2000. *Theory of modeling and simulation: Integrating discrete event and continuous complex dynamic systems*. 2nd Ed., Academic Press, USA

Zulch G, Jonsson U, Fischer J. 2002. Hierarchical simulation of complex production systems by coupling of models. *International Journal of Production Economics* **77**: 39-51