Proceedings of the 2011 Winter Simulation Conference
S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, and M. Fu, eds.

INITIALIZATION OF SIMULATION MODELS USING CMSD

Sören Bergmann
Sören Stelzer
Steffen Straßburger

Ilmenau University of Technology
Helmholtzplatz 3
98684 Ilmenau, GERMANY

ABSTRACT

In the context of online- and symbiotic simulation, the precise initialization of simulation models based on the state of the physical system is a fundamental requirement. In these simulations, the simulation model typically serves as an operational decision support tool. Obviously, it can therefore not start empty and idle. The accurate capturing of initial conditions is fundamental for the quality of the model based predictions. In literature, it is only generally stated that the simulation model must maintain a close connection with the physical system. Our work systematically investigates which data from the physical system is needed for initialization, how it shall be transferred into the simulation model in a standardized way, and which potential problems must be solved in the simulation system to adequately initialize its model elements. We present a solution based on the core manufacturing simulation data (CMSD) standard, suggest necessary extensions and demonstrate a prototypical implementation.

1 INTRODUCTION

Simulation, especially discrete-event simulation, is used within many different disciplines and application areas. In production and logistics, simulation is a well accepted technique for planning and operations (VDI 2000). Simulation is used for the prediction of system behavior and operational decision support, e.g., for the evaluation of control alternatives. To do this, several challenges in modeling and simulation of complex production systems, like the integration of simulation techniques with general manufacturing applications have to be addressed (Fowler and Rose 2004).

A close integration of simulation and general manufacturing applications can also reduce the time between data collection and simulation experimentation; and along this way reduce the occurrence of errors. Techniques like data-driven simulation model generation have been developed in the past to support these objectives (Bergmann and Strassburger 2010, Lorenz and Schulze 1995).

Furthermore, approaches described by the terms online-simulation (Davis 1998) and symbiotic simulation (Aydt et al. 2008) were established aiming at the close integration between a simulation model and the physical system. Online-Simulation was motivated as a support tool for short-term decision making based on state of the physical system. It focuses on predictions of the future behavior of the physical system. Symbiotic simulation extends this to a more "biological" motivated interpretation, and discusses the mutual benefits between the simulation environment and the physical system including the direct control of the physical system.

Both approaches require a mechanism to permanently and quickly "feed" data from the physical system into the simulation environment to have an up to date state representation of the physical system in the model. The accurate capturing of initial conditions is fundamental for the quality of the model based predictions.

Our work investigates systematically which data from the physical system (here: manufacturing systems) is needed for initialization, how it shall be transferred into the simulation model in a standardized way, and which potential problems must be solved in the simulation system to adequately initialize its model elements. We present a solution based on the core manufacturing simulation data (CMSD) standard, suggest necessary extensions and demonstrate a prototypical implementation.

The remainder of this paper is structured as follows: in Section 2 we discuss related work. Section 3 introduces CMSD and discusses its current application. In Section 4, we first introduce the requirements for the initialization of simulation models and classify the input data required from the physical manufacturing system. We then present a solution how to map this data onto the CMSD standard and suggest required enhancements to CMSD.

In Section 5, we document the verification of our approach by using a model generator capable of building a base model and initializing it with data from the physical system. Section 6 critically reviews the achievements and documents possible areas of future work.

2 RELATED WORK

All simulation models in symbiotic or online simulation can typically be treated as non-terminating simulations, i.e., there are initial conditions which are fundamental for obtaining meaningful simulation results. Therefore the "empty" model has to be initialized in a way that the current real system state is equivalent to the model state.

When discussing the initialization of simulation models the quality and availability of input data obviously has the most impact for the correctness of the model (Davis 1998). While the general mechanism of on-line simulation and the quality of input data is discussed in many ways in literature, there are fewer sources that methodically describe how to represent initialization data.

In Hanisch, Tolujew, and Schulze (2005) two basic strategies for model initialization are discussed. The first alternative is to maintain a "parent" simulation, from which simulations clones can be derived. The parent simulation stays in sync with the physical system and is constantly updated with its state.

The second alternative is to generate and initialize the simulation model on demand by using a suitable model representation of the physical system.

Using the first alternative, one first has to create a simulation model and initialize it appropriately. Only afterwards it can be incrementally updated with the state of the physical system, e.g., whenever some measurement data is obtained. This may be problematic when the prediction of the simulation model contradicts a certain measurement. The model must then take measures to adapt to the actual state of the physical system, e.g., by using rollback mechanisms or by interpolating. Additionally, the simulation environment has to supply mechanisms to derive a child model from the parent model. This is something which is rarely found in commercial simulation packages.

The second alternative suggests to automatically generate and initialize the model at the point where it is needed. Here, all information required for initialization must be acquired at the point of creating the simulation model. Although some detailed implementation problems must be addressed, this approach seems suitable for several commercial simulation packages. In this paper we will discuss how this approach can be implemented using a standardized exchange format for the domain of manufacturing systems.

To return to the initially stated objective of reaching a closer integration of simulation with general manufacturing applications, the usage of standards has to be discussed. In the last years, there have been a number of efforts for standardization of exchange formats for model descriptions (e.g., CMSD-IM and SysML (Schönherr and Rose 2009)). These standards were developed to support the exchange of modeling information between non-simulation applications and simulation systems.

Their capabilities have been demonstrated in some real world applications (Johansson et al. 2007; Riddick and Lee 2008). Several CSMD based model generators have been presented in the past, including Bergmann, Fiedler, and Strassburger (2010) and Johansson et al. (2007).

As the focus of the presented scenarios is often the planning phase of a production system, most of these model generation approaches produce an "empty" simulation model, which has to run through a warm-up phase. When addressing the operational phase of a production system the model initialization is of much higher importance, though. We therefore address this issue in detail in the following sections.

3 CORE MANUFACTURING SIMULATION DATA (CMSD) INFORMATION MODEL

The core manufacturing simulation data (CMSD) information model is an open standard developed within the simulation interoperability standards organization (SISO). It has been developed in cooperation with academic and industrial partners under the leadership of the National Institute of Standards and Technology (NIST).

The primary objective of the CMSD Information Model is to facilitate interoperability between simulation systems and other manufacturing applications. Towards this objective it provides a data specification for the efficient exchange of manufacturing data in a simulation environment (SISO 2010). Purposes of the CMSD standard are to:

- enable data exchange between simulation applications and other software applications
- support the construction of manufacturing simulators
- support the testing and evaluation of manufacturing software
- enable greater manufacturing software application interoperability (SISO 2010)

Two different methods are used for representing the CMSD standard: the Unified Modeling Language (UML), and the XML schema definition language (XSD). The UML representation has been organized using packages shown in Figure 1. The classes suitable for the initialization of simulation models are discussed in section 4.2. For more detailed information about CMSD, see SISO (2010), Leong, Lee and Riddick (2006), and Johansson et al. (2007).

The CMSD standard provides data structures and an information model which were designed to firstly support the exchange of modeling information. To cover the complexity of production and logistic systems and a wide range of modeling approaches, the standard allows aspects of the system to be mapped in CMSD in multiple ways.

All CMSD classes which describe objects, like jobs, machines and so on can be extended by user defined properties.

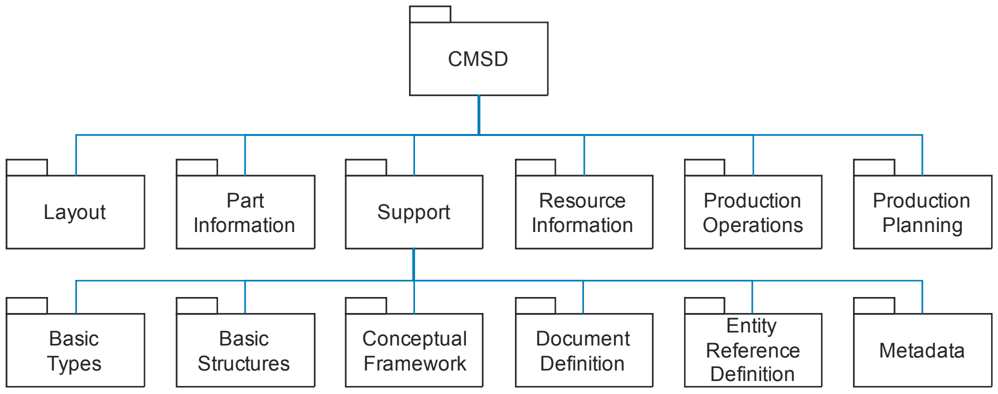


Figure 1: The packages of the CMSD Information Model (SISO 2010)

The capabilities of CMSD were demonstrated in a research project (FACT), which focuses on the developing of new and modified production systems (Johansson et al. 2007). The results of their experiences were used to enhance and evaluate the standardization process. This project was settled in the system planning process. There are no documented applications of using CMSD in system operations. However, the data structures provided by CMSD seem also capable of transporting information of dynamic system components. In this paper we show how it is possible to map state information to CSMD classes so it can be used to initialize simulation models.

Before we describe how the modeling and state information could be mapped into CMSD data structures we first have to discuss and classify the input data of the simulation model obtained from the production or logistics systems.

4 METHODOLOGY FOR MODEL INITIALIZATION

4.1 Input Data

Various data which describe the production system and its state are needed for the simulation studies. The VDI (The Association of German Engineers) classifies relevant input data into three clusters: technical, organizational and system load data as shown in Figure 2.

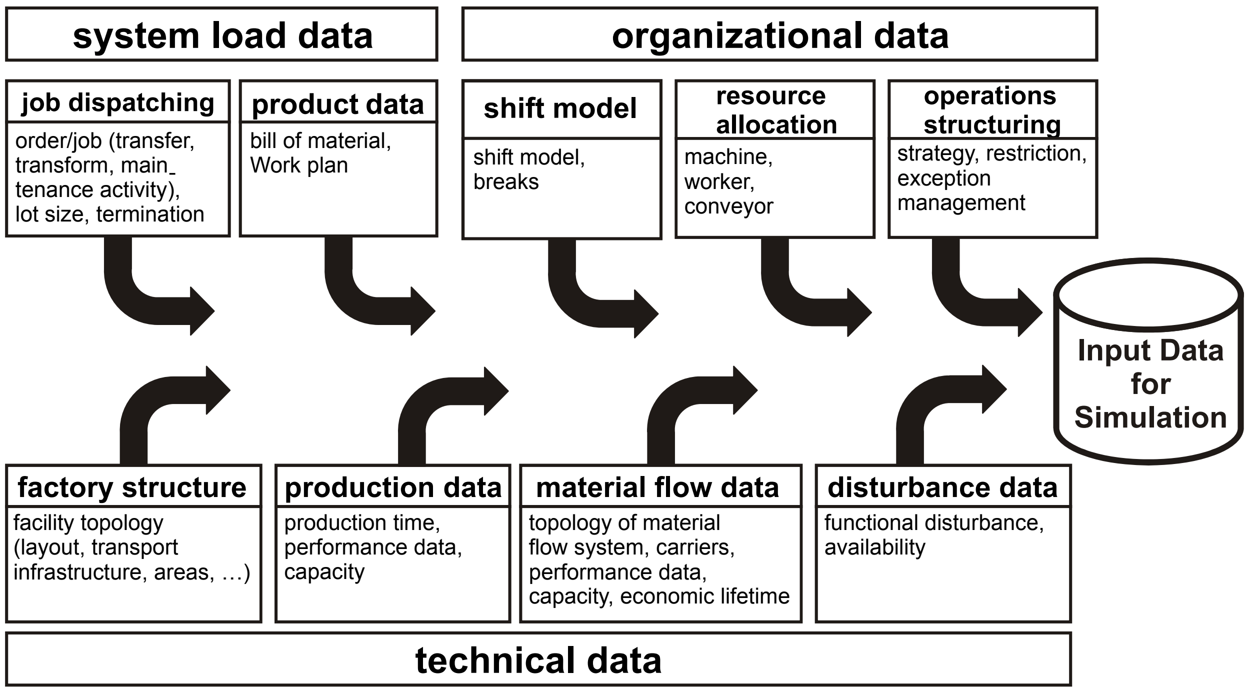

system load data
organizational data
job dispatching
order/job (transfer, transform, main-tenance activity), lot size, termination
product data
bill of material, Work plan
shift model
shift model, breaks
resource allocation
machine, worker, conveyor
operations structuring
strategy, restriction, exception management
Input Data for Simulation
factory structure
facility topology (layout, transport infrastructure, areas, …)
production data
production time, performance data, capacity
material flow data
topology of material flow system, carriers, performance data, capacity, economic lifetime
disturbance data
functional disturbance, availability
technical data

Figure 2: Relevant input data for simulation studies (VDI 2000)

The class of the technical data describes the layout and topology of the entire system as well as the properties of single system components. The organizational data specifies the operation structuring and process organization, especially working shifts models, strategies and resource allocations. Finally, the class system load data describes jobs and their properties.

While technical data and organizational data are mostly relevant for data-driven model generation approaches, the system load data focuses on the data primarily relevant for model initialization.

In this paper we define initialization as a manner to setup simulation models in such a way that the model's internal control structures (event lists, random number generators, simulation clock, component states, etc.) reflect the current state of a real system with sufficient accuracy for forecasting purposes (Hotz 2007).

For the initialization only data about the system load and the state of all resources is of interest (Table 1). We assume that all other aspects, like technical data (system layout, topology) are already reflected in the simulation model, e.g., by using manual or automated model generation approaches (Bergmann and Strassburger 2010).

Table 1: Categories of initialization data

Data about		**Example characteristics**
Resources	Machine status	Idle, working, setup, paused, failed…
	Worker	Place, working, paused, …
	Conveyor	Idle, working, paused, failed. … speed, type, number
Job		Process step, state, scrap percentage, type …
Part		Place, state
System time		

Data on the states of resources is the first class of data which is required to initialize a simulation model. Concerning machines, the active setup of the machine and its current working state are particularly important. Fundamentally, we can distinguish six main working states of a machine: idle, busy, setup, broken/failed, paused and under maintenance. The information which specific job currently occupies the machine is only of secondary interest, as this can typically be modeled as a property of the job.

While machines are typically immobile, we have to distinguish other resources like workers and conveyors, for which the current location can also be of relevance.

Workers have partially other relevant states as other resources. Similar to the machines they have an attribute "working state", but it can have other values. While "in movement" is a valid status for a worker, "failed" is not. Furthermore, workers are usually mobile resources, so they have a current location (often at a machine) and when in movement they should have a destination and an arrival time.

Conveyor is a class of resource which can have quite heterogeneous properties depending on the type of conveyor. Depending on the level of detail in the simulation model, in the simplest case they can be treated like machines. Other parameters, such as current speed, acceleration, type, location and number of carriers can be important if they are represented in the simulation model.

The central element for initialization of simulation models are the jobs in the system, as they represent the dynamic objects of the physical system. Without their accurate reproduction in the model, we cannot use it as a tool for operational decision support. The basic requirement for initializing a job appropriately is to know its current process step and its processing status. It also has to have knowledge about its process plan, e.g., the machine order.

From these two facts crucial information for the simulation can be derived: If a job is at a certain process step (say 7) and has a certain state (say blocked) we can derive that it is located in a buffer in front of machine 7. Similarly, if its state is "started" we can derive that it is being processed at a certain machine.

Furthermore, information about scrap or rework percentages can be relevant, especially when dealing with lot sizes larger than one.

Further detailed information about jobs can be needed when considering parts which are consumed, e.g., in assembly processes. Here we might need data on availability of parts at the current workstation.

From the states and conditions discussed above, a certain set can be used for initialization quite easily. This is especially true for all enumerated data types which merely describe a state of an element (e.g., machine state "idle").

Other data, like the current status of already started jobs (including maintenance or repair jobs) can be quite difficult to capture from the real system and to map into the simulation model state. First of all, this data will most likely not be explicitly available from the real system. Rather, if we want to know a remaining process time, we will most likely only be able to determine a job's starting time and its planned processing time. From this we may be able to estimate its remaining processing time. Still, it may be difficult to appropriately integrate this information into the simulation system.

4.2 Using CMSD for Modeling Input Data

The CMSD standard offers a variety of classes which can be used for representing the data discussed in the previous section. In this section we focus on the discussion of those CMSD classes which seem appropriate for model initialization purposes. In particular, we suggest the usage of the classes Resource, Part, Job, JobEffortDescription, Schedule, ScheduleItem and ProcessPlan (see Figure 3).

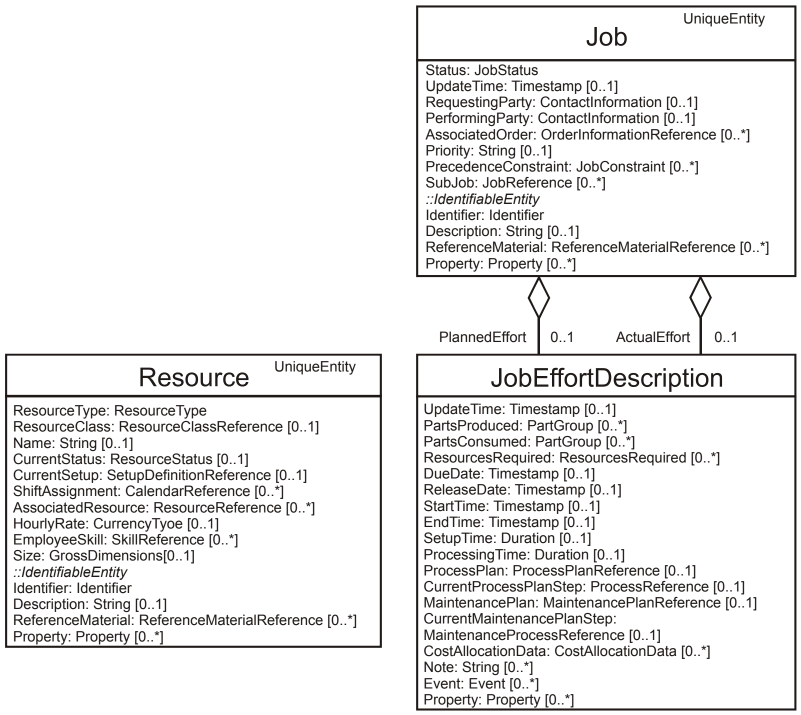

Job
UniqueEntity
Status: JobStatus
UpdateTime: Timestamp [0..1]
RequestingParty: ContactInformation [0..1]
PerformingParty: ContactInformation [0..1]
AssociatedOrder: OrderInformationReference [0..*]
Priority: String [0..1]
PrecedenceConstraint: JobConstraint [0..*]
SubJob: JobReference [0..*]
::IdentifiableEntity
Identifier: Identifier
Description: String [0..1]
ReferenceMaterial: ReferenceMaterialReference [0..*]
Property: Property [0..*]
PlannedEffort 0..1
ActualEffort 0..1
Resource
UniqueEntity
ResourceType: ResourceType
ResourceClass: ResourceClassReference [0..1]
Name: String [0..1]
CurrentStatus: ResourceStatus [0..1]
CurrentSetup: SetupDefinitionReference [0..1]
ShiftAssignment: CalendarReference [0..*]
AssociatedResource: ResourceReference [0..*]
HourlyRate: CurrencyTyoe [0..1]
EmployeeSkill: SkillReference [0..*]
Size: GrossDimensions[0..1]
::IdentifiableEntity
Identifier: Identifier
Description: String [0..1]
ReferenceMaterial: ReferenceMaterialReference [0..*]
Property: Property [0..*]
JobEffortDescription
UpdateTime: Timestamp [0..1]
PartsProduced: PartGroup [0..*]
PartsConsumed: PartGroup [0..*]
ResourcesRequired: ResourcesRequired [0..*]
DueDate: Timestamp [0..1]
ReleaseDate: Timestamp [0..1]
StartTime: Timestamp [0..1]
EndTime: Timestamp [0..1]
SetupTime: Duration [0..1]
ProcessingTime: Duration [0..1]
ProcessPlan: ProcessPlanReference [0..1]
CurrentProcessPlanStep: ProcessReference [0..1]
MaintenancePlan: MaintenancePlanReference [0..1]
CurrentMaintenancePlanStep:
MaintenanceProcessReference [0..1]
CostAllocationData: CostAllocationData [0..*]
Note: String [0..*]
Event: Event [0..*]
Property: Property [0..*]

Figure 3: Resource and job related classes (SISO 2010)

In the following discussion we assume that the simulation model generation has created all necessary elements, parameters, and data types consistent with potential initialization values (e.g., all actually occurring setup states must be present and meaningful to the model).

We first describe the set of basic initialization data (Table 2) that can be mapped directly onto attributes of the classes distinguished in CMSD (figure 4). Later in this section, we will also describe extensions needed in CMSD and how to implement them using the property concept of CMSD.

Machines as the most important resource type can be completely described in terms of their current setup and their operational readiness. The current setup state is represented as a reference to a setup state definition, which was created during the model generation. The operational readiness can be described through an enumeration value of the ResourceStatus class. The predefined enumeration values (busy, idle, broken, under maintenance, and unknown) are not sufficient. We suggest to include at least the states "paused" and "in setup."

The basic state of workers and conveyors can be described analog to machines, with the difference that workers have no setup state and some ResourceStatus does not apply (e.g., broken).

Next to resources, jobs are the most important elements for accurate model initialization. In CMSD, the Schedule and the ScheduleItem classes have to be used to specify scheduled jobs that are not yet released into the production system. The classes Job and JobEffortDescription are used for Jobs which are already in the production system. They define the current or the next step of a job's process plan and all actual and planned times/dates. A job can be a normal manufacturing, a maintenance or a repair operation. Again it is important that the simulation model is consistent with the intended initialization, e.g., all necessary process plans must be available.

As shown, the classes as defined in the CMSD standard are sufficient for basic initialization purposes (Table 2). For applications requiring a more detailed modeling and initialization, some extensions will be needed which can be brought into CMSD as a proprietary extension using the CMSD property concept.

Table 2: Used CMSD classes and attributes

Data	**CMSD Class**	**Relevant Attributes**
Machine state	Resource (type = machine or station)	CurrentStatus: ResourceStatus
		AssociatedResource: ResourceReference (Worker)
Worker	Resource (type = employee)	CurrentStatus: ResourceStatus
		Property - current location (LocationDefinition)
Conveyor	Resource (type = carrier, conveyor, “power and free” or transporter)	CurrentSetup: SetupDefinitionReference
		CurrentStatus: ResourceStatus
		AssociatedResource: ResourceReference (Worker)
		Property - current speed, acceleration, and type, location and number of carrier
Job	Job	Status: JobStatus
		Priority. String
		ActualEffort: JobEffortDescription
		PlannedEffort: JobEffortDescription
	JobEffortDescription	DueDate / ReleaseDate: TimeStamp
		StartTime / EndTime: TimeStamp
		ProcessPlan: ProcessPlanReference
		CurrentProcessPlanStep: ProcessReference
		MaintenancePlan: MaintenancePlanReference
		CurrentMaintenancePlanStep: MaintenanceProcessRef.
		Property - remaining processing times (double) [%]
Schedule	Schedule	StartTime / EndTime: Timestamp
		ScheduleItem: ScheduleItem
	ScheduleItem	AssociatedJob: JobReference
Part	Part	ProductionStatus: PartProductionStatus
		Location: LocationDefinition
System time	HeaderSection	CreationTime: Timestamp

As discussed in section 4.1, the detailed processing status of started jobs is certainly the information that adds the most value to many simulation scenarios as it increases the accuracy of the initialization. We therefore suggest to map the remaining processing time to a simple CMSD property, named "RemainingProcessingTime." It is conceivable that either the percentage and/or the absolute remaining processing time is specified. In our test implementation (see Section 5) we use percentage values.

Another suggested extension is the inclusion of the number of parts of a lot that are scrapped or need to be reworked. Again, these parameters can be specified in percent of the lot or as an absolute number of units.

Further extensions based on the CMSD property concept are possible but they highly depend on the specific manufacturing scenario. While most of these extensions can be easily modeled using CMSD properties, this holds the danger of creating incompatible CMSD dialects.

Compared to other resources like machines or workers, conveyer systems are not so easy to abstract and therefore take a separate discussion. In CMSD conveyers are primarily resources, which are described by static attributes. In reality, however, conveyers are accumulating objects that can hold and buffer work items. In our modeling approach these work items are identified by job objects. Unfortunately, CSMD does not provided attributes to describe the current state of the capacity of a conveyer system.

```
<ProcessPlan>
      <Identifier> ProcessPlan01_PartType01 </Identifier>
      <PartsProduced>
            <PartType> <PartTypeIdentifier> PartType01 </PartTypeIdentifier> </PartType>
      </PartsProduced>
      <Process>
            <Identifier> PP01_PartType01_Step01 </Identifier>
            <ResourcesRequired>
                  <Resource> <ResourceIdentifier> Ma02 </ResourceIdentifier> </Resource>
                  <AllowableSetup>
                        <SetupDefinitionIdentifier> Setup_Ma02_PartType01 </SetupDefinitionIdentifier>
                  </AllowableSetup>
            </ResourcesRequired>
            <OperationTime> <TimeUnit> minutes </TimeUnit> <Value> 35 </Value> </OperationTime>
      </Process>
      <Process/> ... <Process/>
<ProcessPlan>
```

```
<Job>
      <Identifier> Job02 </Identifier>
      <Status> started </Status>
      <UpdateTime> 2011-06-15T16:23:16Z </UpdateTime>
      <PlannedEffort>
            <StartTime> 2011-06-14T10:23:00Z </StartTime><EndTime> 2011-06-24T11:31:00Z </EndTime>
            <ProcessPlan>
                  <ProcessPlanIdentifier> ProcessPlan01_PartType02 </ProcessPlanIdentifier>
            </ProcessPlan>
            <CurrentProcessPlanStep>
                  <ProcessIdentifier> PP01_PartType02_Step02 </ProcessIdentifier>
            </CurrentProcessPlanStep>
      </PlannedEffort>
      <Property>
            <Name> RemainingProcessTime </Name> <Unit> minutes </Unit> <Value> 25.4 </Value>
      </Property>
</Job>
```

Figure 4: Snapshot of the XML-based CMSD description for Job- and Process-Definition

There are two ways to solve this limitation. The first alternative is to use again the property concept and extend the resource class with capacity state information. We suggest using the second alternative, which is to model the time consumption of jobs while transported by a conveyer system in its attached process plan. By processing the process plans of jobs, the currently used resource can be determined. With this information the current content state of a conveyor can be calculated. This kind of modeling could also be used for buffers.

While the suggested extensions using properties are designed to increase the accuracy of initializing simulation models, a backward compatibility is maintained, as initialization routines not capable of handling a certain property will still be able to perform basic initialization, even if it is at a lower degree of accuracy.

5 TEST IMPLEMENTATION

Our pilot implementation is built on an existing CSMD based model generator created in previous research work by Bergmann, Fiedler, and Strassburger (2010).

For our test environment (Figure 5) we have designed the information flow as closely as possible to real world scenarios. We assume that all information which is required to describe the system and its state is captured in the relevant manufacturing applications (e.g., Enterprise Resource Planning Systems (ERP) and Manufacturing Execution System (MES)).

We emulate that process by creating a valid CSMD description of the system. In our prototype, the CMSD XML files are generated by a web based frontend. In reality, these files would be extracted from the manufacturing applications.

Our model generation and initialization approach supports incremental model generation, i.e., static components described in CMSD only need to be generated once, while initialization can be performed multiple times as required.

The generated CSMD description contains every information which is necessary to generate the simulation model of the physical system and initialize it with its current state. This information package is sent to our CMSD based model generator.

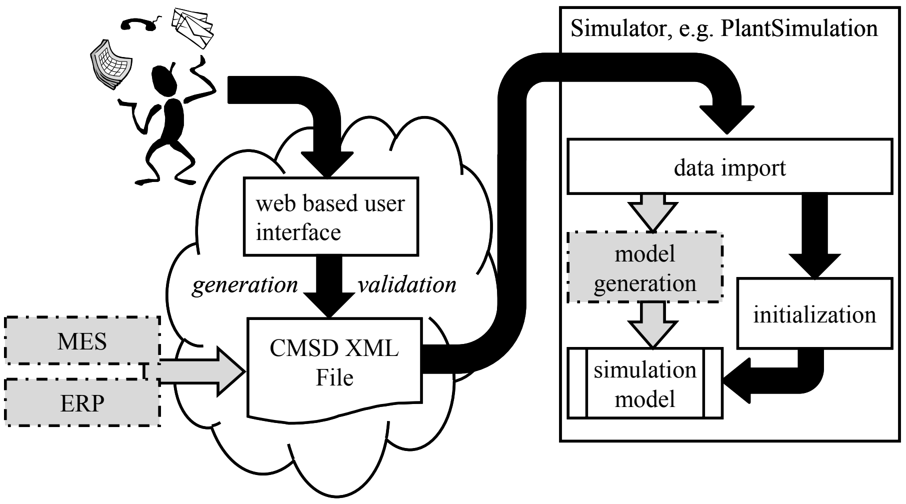
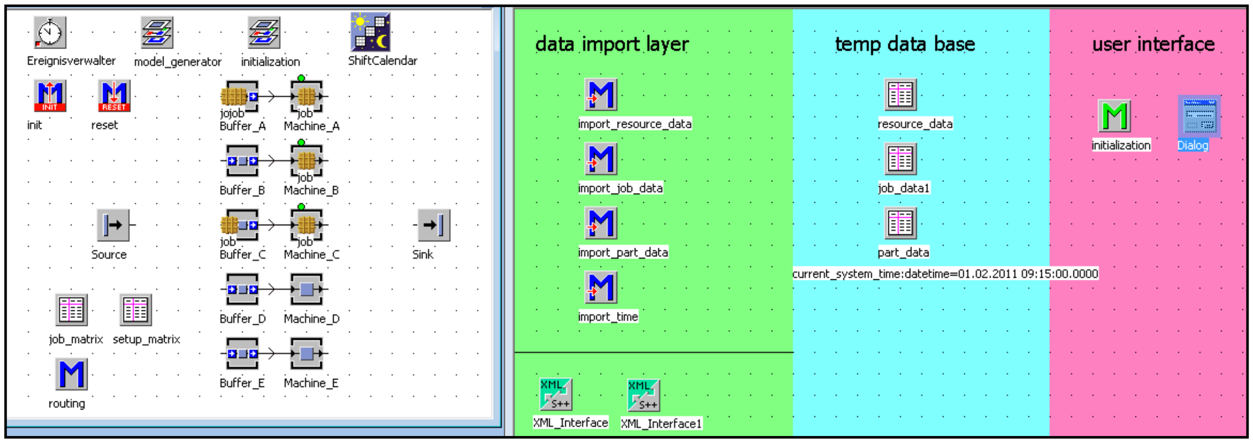


Figure 5: Prototype for automatic generation and initialization of simulation models

The model generator itself serves as a transformation layer between CMSD and the used simulation environment. It processes the information contained in the CMSD description and maps it to internal structures and components of the simulation environment. The challenge is to ensure that the simulation system is actually capable of representing the physical system and allowing its initialization.

From the large number of commercial simulation packages available not every simulation environment can fulfill these requirements. For our implementation we have used Plant Simulation (Siemens 2011) as simulation environment which matches these requirements (Figure 6).



current_system_time:datetime=01.02.2011 09:15:00.0000

Figure 6: Plant Simulation prototype for CMSD based initialization

The test scenario depicts a typical job-shop production, with 5 machines and 5 job types. The details are shown in Tables 3 and 4. Each machine has an input buffer with unlimited capacity. For simplicity, the processing strategy of the buffer is in the order of the storage (first in first out - FIFO).

Table 3: Scenario job matrix

Job type	**Machine order**	**Processing time (in minutes)**
A	M1, M2, M3, M4, M5	3, 5, 6, 4, 2
B	M2, M3, M1, M4	2, 5, 4, 3
C	M2, M1, M4, M5, M3	3, 4, 5, 1, 4
D	M1, M2, M3, M4	2, 3, 5, 3
E	M2, M3, M5	2, 4, 6

Table 4: Scenario setup matrix (left) and machine parameters (right)

Job type	**A**	**B**	**C**	**D**	**E**
A	0	1	2	1	3
B	1	0	3	1	1
C	2	3	0	2	2
D	1	1	2	0	1
E	3	1	2	1	0

Machine	**MTBF**	**MTTR**
M1	15	2
M2	15	5
M3	20	5
M4	25	2
M5	15	5

For the described model we derived several test cases with information for various states. For every case we investigate the result of the model generation process for completeness concerning information mapping, generating correct simulation models and accuracy of the model behavior. We also investigated the performance of the model generator, especially the time to build and initialize simulation models from scratch. We could determine that the speed of the model generation process depends highly on the XML processing time, which directly depends on the size of the CMSD description.

In summary, our model generator is able to generate simulation models and initialize them with the initialization data specified in Section 4.2. It is also capable of performing initialization without rebuilding the model from scratch. In this case the XML processing time is significantly decreased.

6 CONCLUSION AND FUTURE WORK

The goal of our work was to investigate requirements and solutions for simulation model initialization in the context of online- and symbiotic simulation in the domain of manufacturing simulation. In these types of simulations, the simulation model typically serves as an operational decision support tool. The accurate capturing of initial conditions is therefore fundamental for the quality of the model based predictions.

Our work investigated systematically which data from the physical system is needed for initialization focusing on the domain of manufacturing simulation. Based on these findings we have investigated the CMSD standard for its capabilities to carry model initialization information. While CMSD has been shown to be generally capable of carrying basic initialization information, extensions will be needed for more specific initialization tasks. We have further shown how the CMSD property concept can be used to implement these extensions.

We have further demonstrated a prototypical implementation of a CMSD based model generator capable of generating and initializing a simulation model in Plant Simulation. We are convinced that the results of our research represent a viable approach for model initialization that can be used to improve applications where a large amount of simulation experimentation has to be performed, like symbiotic simulation, simulation based early warning systems or model predictive control. Due to the generic design of the model generator we are able to support cyclic model generation which facilitates the adaption of the entire simulation model when the real system changes.

Future work will involve the testing of the approach in larger industrial test scenarios. Potential directions of future work include the topic of providing simulation (including model generation and initialization) as a service and offering the large amount of scenarios to be analyzed in the context of symbiotic simulation as cloud-based simulation services.

REFERENCES

Aydt, H., S. J. Turner, W. Cai, and M. Y. H. Low. 2008. "Symbiotic simulation systems: An extended definition motivated by symbiosis in Biology." In *Proceedings of the 22nd Workshop on Principles of Advanced and Distributed Simulation*, edited by S. Ceballos, 109–116. Rome, Italy: Institute of Electrical and Electronics Engineers, Inc.

Bergmann, S., A. Fiedler, and S. Straßburger. 2010. "Generierung und Integration von Simulationsmodellen unter Verwendung des Core Manufacturing Simulation Data (CMSD) Information Model" (Generation and integration of simulation models using the Core Manufacturing Simulation Data (CMSD) information model (in German)). In *Proceedings of the 14th ASIM Dedicated Conference on Simulation in Production and Logistics - Integration Aspects of Simulation Referring to Equipment, Organization and Personne*l, edited by G. Zülch and P. Stock, 461-468. Karlsruhe Institute of Technology (KIT).

Bergmann, S., and S. Strassburger. 2010. "Challenges for the Automatic Generation of Simulation Models for Production Systems." In *Proceedings of the 2010 Summer Simulation Multiconference*, 545-549. Ottawa, Canada, July 11-14.

Davis, W. J. 1998. "On-Line Simulation: Need and Evolving Research Requirements." In *Handbook of Simulation*, edited by J. Banks, 465-516. John Wiley & Sons, Inc.

Fowler, J. W., and O. Rose. 2004. "Grand Challenges in Modeling and Simulation of Complex Manufacturing Systems." *SIMULATION* 80(9):469–476.

Hanisch, A., J. Tolujew, and T. Schulze. 2005. "Initialization of Online Simulation Models." In *Proceedings of the 2005 Winter Simulation Conference*, edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, 1795-1803. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Hotz, I. 2007. "Ein Simulationsbasiertes Frühwarnsystem zur Unterstützung der operativen Produktionssteuerung und -planung in der Automobilindustrie" (A simulations based early warning system to support the operational production planning and scheduling in the automotive industry (in German)). Ph.D. thesis, Otto-von-Guericke University Magdeburg, Accessed April 1, 2011. http://diglib.uni-magdeburg.de/Dissertationen/2007/inghotz.pdf.

Johansson, M., S. Leong, Y. T. Lee, F. Riddick, G. Shao, B. Johansson, A. Skoogh, and P. Klingstam. 2007. "A Test Implementation of the Core Manufacturing Simulation Data Specification." In *Proceedings of the 2007 Winter Simulation Conference*, edited by S. G. Henderson, B. Biller, M.-H Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 1673-1681. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Leong, S., Y. T. Lee, and F. Riddick. 2006. "A Core Manufacturing Simulation Data Information Model for Manufacturing Applications". In: *Proceedings of the 2006 Fall Simulation Interoperability Workshop*. Simulation Interoperability Standards Organization. Orlando, Florida.

Lorenz, P., and T. Schulze. 1995. "Layout Based Model Generation." In *Proceedings of the 1995 Winter Simulation Conference*, edited by C. Alexopoulos, K. Kang, W. R. Lilegdon, and D. Goldsman, 728-735. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Riddick, F., and Y. T. Lee. 2008. "Representing Layout Information in the CMSD Specification." In *Proceedings of the 2008 Winter Simulation Conference*, edited by S. J. Mason, R. R. Hill, L. Mönch, O.

Rose, T. Jefferson and J. W. Fowler, 1777-1784. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Schönherr, O., and O. Rose. 2009. "First Steps Towards a General SysML Model for Discrete Processes in Production Systems." In *Proceedings of the 2009 Winter Simulation Conference*, edited by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, 1711-1718. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Siemens. 2011. "Plant Simulation." Siemens Product Lifecycle Management Software Inc. Accessed March 28, 2011. http://www.plm.automation.siemens.com/en_us/products/tecnomatix/plant_design/plant_simulation.shtml.

SISO. 2010. "Standard for: Core Manufacturing Simulation Data – UML Model." Core Manufacturing Simulation Data Product Development Group, Simulation Interoperability Standards Organization. Accessed February 14, 2011. http://www.sisostds.org/DigitalLibrary.aspx?Command=Core_Download&EntryId=31457.

VDI. 2000. "VDI 3633-1: Simulation of Systems in Materials Handling, Logistics and Production – Fundamentals." VDI-Society Production and Logistics. Berlin: Beuth Verlag.

AUTHOR BIOGRAPHIES

SÖREN BERGMANN is a PhD student at Ilmenau University of Technology. He is a member of the scientific staff at the Department for Industrial Information Systems. He received his diploma degree in Information Systems from Ilmenau University of Technology. Previously he worked as corporate consultant in various projects. His research interests include generation of simulation models and automated validation of simulation models within the digital factory context. His email is soeren.bergmann@tu-ilmenau.de.

SÖREN STELZER is a PhD student at Ilmenau University of Technology. He is a member of the scientific staff at the Department for Industrial Information Systems. He received his diploma degree in Computer Science from the Ilmenau University of Technology. During his study he was working in the Neuroinformatics and Cognitive Robotics Lab of the Ilmenau University of Technology. After his study he was working in optimization of power plants in several projects. His research interests are simulation based optimization, model predictive control, artificial learning and discrete event simulation. His email is soeren.stelzer@tu-ilmenau.de.

STEFFEN STRASSBURGER is a professor at the Ilmenau University of Technology in the School of Economic Sciences. Previously he was working as head of the "Virtual Development" department at the Fraunhofer Institute in Magdeburg, Germany and as a researcher at the DaimlerChrysler Research Center in Ulm, Germany. He holds a Ph.D. and a Diploma degree in Computer Science from the University of Magdeburg, Germany. He is a member of the editorial board of the Journal of Simulation. His research interests include distributed simulation as well as general interoperability topics within the digital factory context. He is also the Vice Chair of SISO's COTS Simulation Package Interoperability Product Development Group. His web page can be found via www.tu-ilmenau.de/wi1. His email is steffen.strassburger@tu-ilmenau.de.