Facility Logistics Systems Modeling

Leon McGinnis
Edward Huang
Kysang Kwon
Georgia Institute of Technology

Abstract

While the facility logistics research literature is replete with examples of models to support design and operational decisions, there has been little progress in model integration, particularly in the form of integrated computational support. In this paper we describe a new systems modeling language—SysML—and how it has been used to prototype a factory design process which integrates standard authoring tools for describing processes, process plans, factory layouts, and production requirements, and provides on-demand generation of high fidelity factory simulations. By formalizing the semantics of the factory, SysML enables a high degree of knowledge capture and re-use. Because it is based on UML, SysML supports very sophisticated computational manipulation of models. Together, these two key attributes provide a basis for unifying facility logistics models in a computational form relevant for both research and practice.

1. Introduction

Over the past twenty years, the International Material Handling Research Colloquium has played a major role in transforming material handling research, by bringing together and challenging researchers and practitioners to identify and address the most important unsolved problems in the field. One element of the IMHRC legacy is the rich and growing archival literature on modeling and analysis of MHS operations (see, e.g., Gu *et al*, 2007 on warehouse operations models). The predominant theme in the archival literature is the modeling and analysis of specific operations (e.g., transporting lots in a factory, picking orders in a warehouse), or specific technical subsystems (automated guided vehicles, rotary racks, AS/RS, order accumulation/sortation, etc). There is little archival literature that addresses integrated modeling of facility logistics systems from a formal or quantitative perspective. This represents a significant hurdle to the deployment of existing research results in practice, and a major opportunity for research.

The work reported in this paper is based on the following hypothesis: the relative paucity of research on integrated, comprehensive systems modeling in facility logistics is due to the historic lack of a sufficiently powerful and expressive systems modeling language that could be used to integrate the extant operational and design models. If such a language was available and was used to create integrated comprehensive facility logistics systems models, it would validate the hypothesis, and point the way to a new generation of facility logistics research.

In May, 2006, OMG (Object Management Group) published an initial standard for a new modeling language, which was designated "Systems Modeling Language," or SysML. SysML is based on the Unified Modeling Language (UML), which is widely used in software engineering, and software systems design and development. While there have been a few attempts by engineers to use UML, it has not caught on, probably because of its intense software focus. SysML was specifically designed to support the systems engineering process, especially the specification and design of systems composed of hardware, software, and people. Since SysML is implemented as a profile of UML, development of SysML application software has been relatively rapid. As of January 1, 2008, there were at least seven commercial off-the-shelf (COTS) sources for SysML software (see www.omgsysml.org for up to date listing of sources).

The Product and Systems Lifecycle Management Center (PSLM Center) and the Keck Virtual Factory Lab (VFL) at Georgia Tech have been early adopters of SysML, have used SysML in a variety of projects, and are engaged in the language standards development process. Two relevant projects using SysML are addressing, respectively, semiconductor factory design, and warehouse design. Preliminary descriptions of these projects have been presented in (Huang *et al*, 2007; Kwon *et al*, 2007; McGinnis *et al*, 2006; and McGinnis, 2007)

The goal of this paper is to provide a vision for how a powerful systems modeling language can transform both research and practice in the field of facility logistics, and to support that vision with concrete examples from our research experience. To do this, we will give a brief introduction to semiconductor manufacturing, and illustrate SysML by using SysML diagrams in that introduction. Next we will describe how SysML models of semiconductor manufacturing can be used to support semiconductor factory design. Specifically, we will discuss how SysML has been used to define libraries for FactoryCAD™, and eM-Plant™, to define schema for essential domain data, and to support on-demand generation of high-fidelity factory simulation models. There are a number of implications we draw from this experience, and we conclude the paper with a discussion of challenges for the facility logistics research community.

2. Vision

Designing facility logistics systems should be no different than designing any other large scale complex system, such as a microprocessor or a large passenger aircraft. It should enable a number of disciplines to collaborate in authoring a system description, perhaps with computational support for optimizing some decisions, and it should engage appropriate solvers to provide analyses of the system as described. The capabilities for system description and analysis should span a spectrum of fidelity, from high level (and low fidelity) system descriptions to very low level (and high fidelity) descriptions.

Many different systems perspectives may be represented in the various analyses supporting facility logistics system design. For example, one perspective may focus on assessing processing capacity in a proposed design, one on the response time of the material handling system, and one on the cycle time of products flowing through the system. Each of these perspectives, and others, will be important to the system designers at some point, and each should be available, *on demand*.

Finally, the facility logistics system designer should be a domain expert, i.e., one who is competent to make facility logistics system design decisions, based on both knowledge of the domain, and analysis of specific design requirements and design alternatives. However, the facility logistics system designer should be neither expected nor required to be an expert in each of the models and associated solvers which provide information about requirements or design alternatives. These models and their analyses should be available to the facility logistics systems designer, not as the result of off-line work by modeling and analysis experts, but on-line, *on demand*, as needed to support design decision-making.

3. SysML Overview

"The OMG Systems Modeling Language **(OMG SysML™)** is a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that may include hardware, software, information, personnel, procedures, and facilities" [OMG 2007]. SysML is implemented as a profile of UML [OMG 2007], so it uses the basic modeling approach and many of the modeling constructs of UML, but adds some new modeling constructs specific to systems modeling. Figure 1, taken from [OMG 2007], summarizes the diagrams available in SysML.

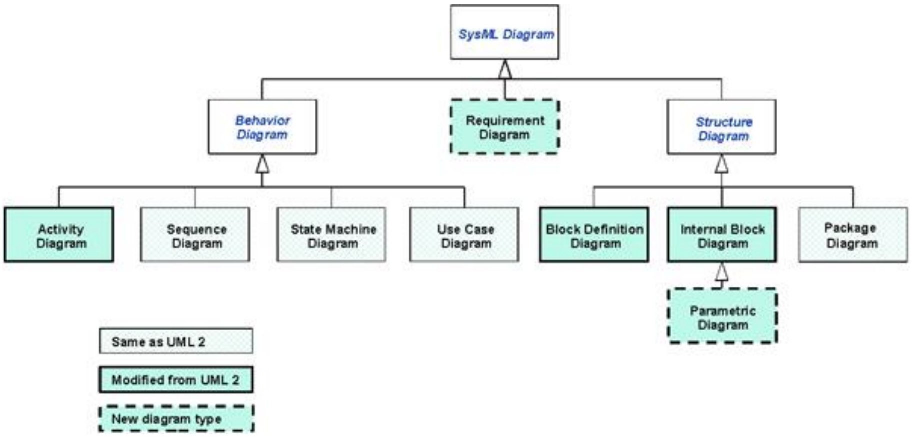


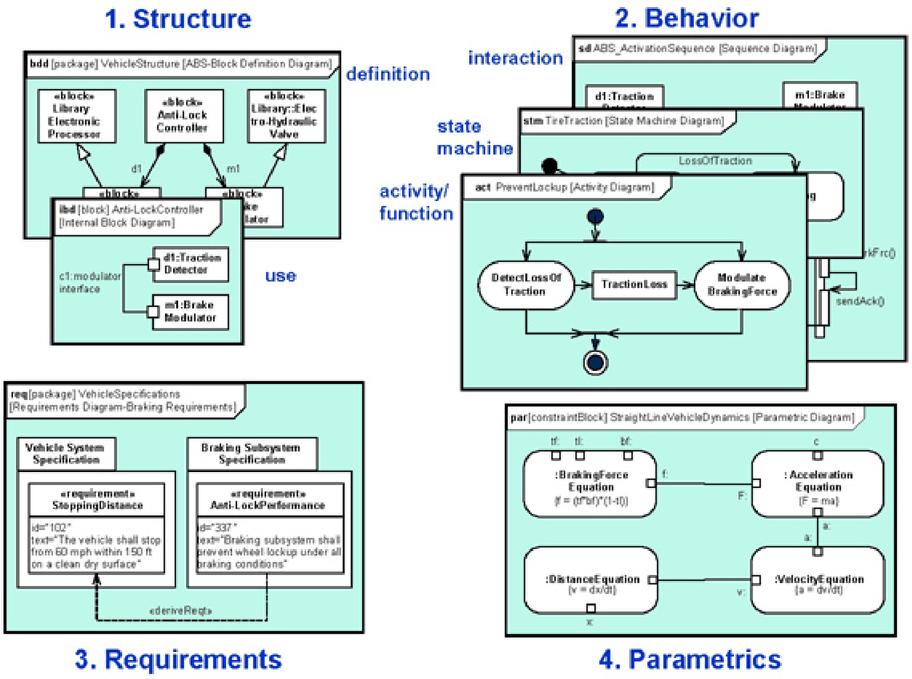
Figure 1. SysML Diagram Types

The graphical language of SysML enables four specific kinds of semantic content in the description of systems:

1. structure: the entities in the system and their relationships;
2. behavior: how entities change, and the types of interactions among entities;
3. requirements: the purpose of the system, what it must accomplish; and
4. parametrics: diagrams that represent "constraints on system property values such as performance, reliability, and mass properties, and serves as a means to

> integrate the specification and design models with engineering analysis models" [OMG 2007].

This is illustrated in figure 2, also taken from [OMG 2007]. Requirements diagrams provide a mechanism for managing the formal requirements that drive system design. Structure and behavior diagrams provide a formal mechanism for describing the system and communicating that description to system implementers, system users, system maintainers, and system analysts. Parametric diagrams provide a mechanism for capturing important parametric relationships in a way that is readily communicated, but more importantly, can be made computational.



Note that the Package and Use Case diagrams are not shown in this example, but are respectively part of the structure and behavior pillars

Figure 2. Four Semantic Capabilities of SysML

4. Semiconductor Manufacturing and SysML

Semiconductor manufacturing is one of the most demanding and complex manufacturing enterprises. A new wafer fab can cost in excess of US$3 billion, can contain over a thousand process tools, dozens of automated storage/retrieval systems, hundreds of transport vehicles, and many kilometers of overhead transport guidepath. Products do not flow "through" a wafer fab in a linear fashion, but repeat a set of process steps over and over again. In total, a wafer may require over a thousand process steps, as it cycles through the factory to receive as many as thirty layers of circuitry and insulation. A wafer may remain in process in the factory for as long as six weeks, and may travel as much as a hundred kilometers. The latest generation of wafer fabs come as close as any factory to the notion of "lights out" operation, with minimal "touch labor" involved in the manufacturing process. The front-opening unified POD, or FOUP, is the container in which wafers are transported automatically from automated station to automated station.

In this section, we will illustrate the use of SysML to describe a wafer fab. What we will describe is, in effect, a "meta-model" of semiconductor manufacturing, and we will later describe how this meta-model is used to create a description of a specific factory. Our objective is not to provide a tutorial on SysML, but simply to illustrate that the essential conceptual model of a wafer fab can be captured using the syntax of SysML. For a thorough introduction to SysML, see [OMG 2007].

Fab Structure Modeling

In describing fab structure, we will address three elements: the physical components of the fab, the organization of the physical components, and the logical components.

Physical structural elements

There are five *classes* of physical structural elements required in describing a wafer fab:

- FOUPs, which are the containers in which wafers are moved; a FOUP may be empty, or may contain any number of wafers from 1 up to its capacity;
- Tools, which are the "workstations" at which device fabrication or metrology operations are performed on the wafers; tools may be further classified, e.g., as "single wafer tools" or as "batch tools" or in other useful ways; tools have "ports" to which FOUPs attach for the transfer of wafers in and out of the tool.
- Stockers, which are a special category of tool; a stocker is an automated storage/retrieval system which provides temporary storage for a FOUP when the next operation requires a tool that is not currently available, or if the lot contained in the FOUP is being held for some reason.
- Vehicles, which are overhead transport vehicles capable of transporting one FOUP at a time, and can be routed through the available transport network to provide point-to-point delivery of a FOUP; these vehicles have the properties of AGVs, in that they incorporate some form of collision avoidance, and may require contention resolution at intersections
- Path segments, which are the physical components of the transport network; path segments may be curved or straight, or may represent merge/diverge branches, or intersections where guidepath segments cross.

In SysML, each of these classes is represented as a *block*, which may have any number of properties; these properties may represent characteristic data, a link to a drawing file, or any other information that is relevant to connect with the class.

Like UML, SysML is object-oriented, so relationships between blocks, such as inheritance may be represented in the SysML diagram. Figure 3 illustrates the description of the tool class, where a tool has one or more ports, and a stocker is a special type of tool. Figure 4 illustrates the description of an AMHS, which has one or more segments in a network, which itself consists of "arcs" and "move points." Arcs are further refined into straight and curved arcs, and move points further refined into interface points and branching points. In figure 4, a special case of "AMHS" is the "interbay AMHS" which transports FOUPS between bays in the fab. The interbay AMHS has one or more OHT vehicles.

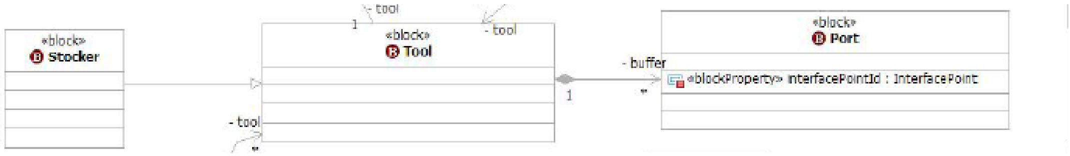


Figure 3. Tools, Ports, and Stockers

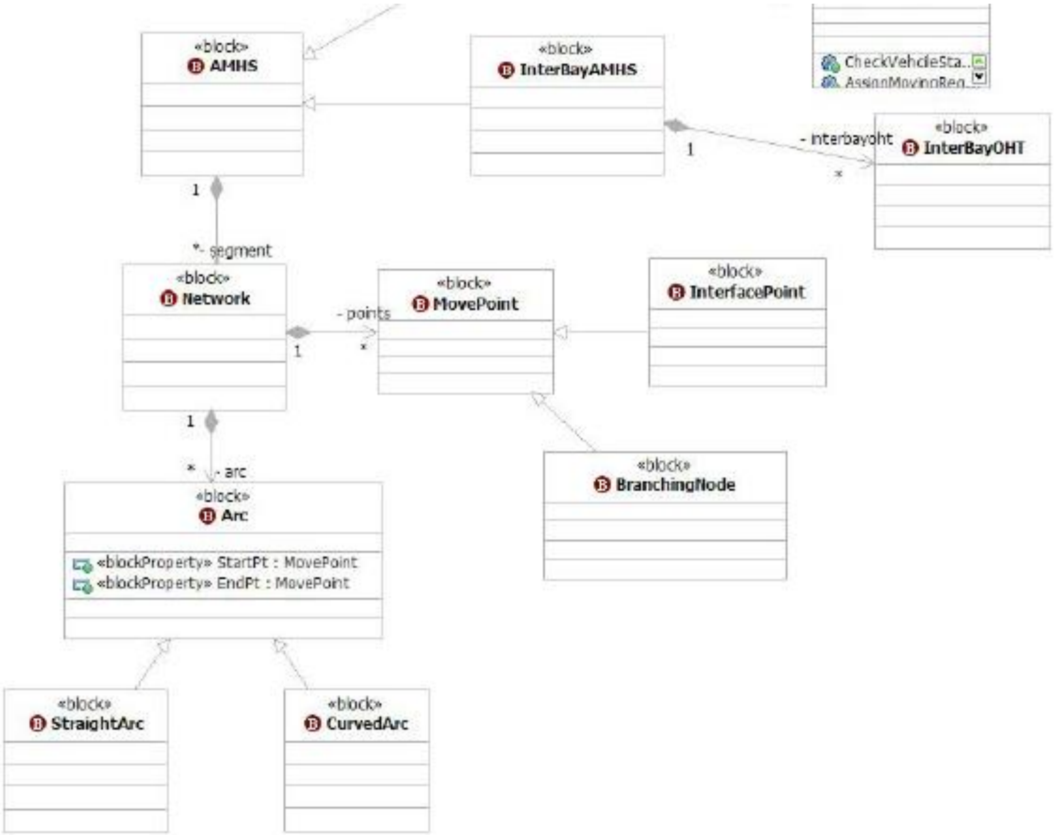


Figure 4. AMHS components

Physical Organization

The resources in a fab will have some physical organization. Here, we describe what is commonly referred to as a "bay layout with through-stocker transport." Bays might refer to a physical configuration of the fab building, where a "chase" is used to isolate tools from the "clean room" in which the wafers are transported. With the advent of

FOUPS, the chase is no longer a physical requirement, and bays may simply represent an organization of tools into “departments” in an attempt to achieve efficiencies. Traditionally, bays have their own intrabay AMHS, and FOUPS enter the bay through a stocker, or AS/RS, which connects the bay to the interbay AMHS. Figure 5 contains a portion of the block definition diagram, or BDD, describing these relationships. A Bay has an intrabay AMHS, which has one or more OHTs, and one or more job queues. The Intrabay AMHS is a specialization of the generic AMHS, which means it has a path network consisting of both interface and move points. Some of those move points sill correspond to tool ports, and some to stocker ports.

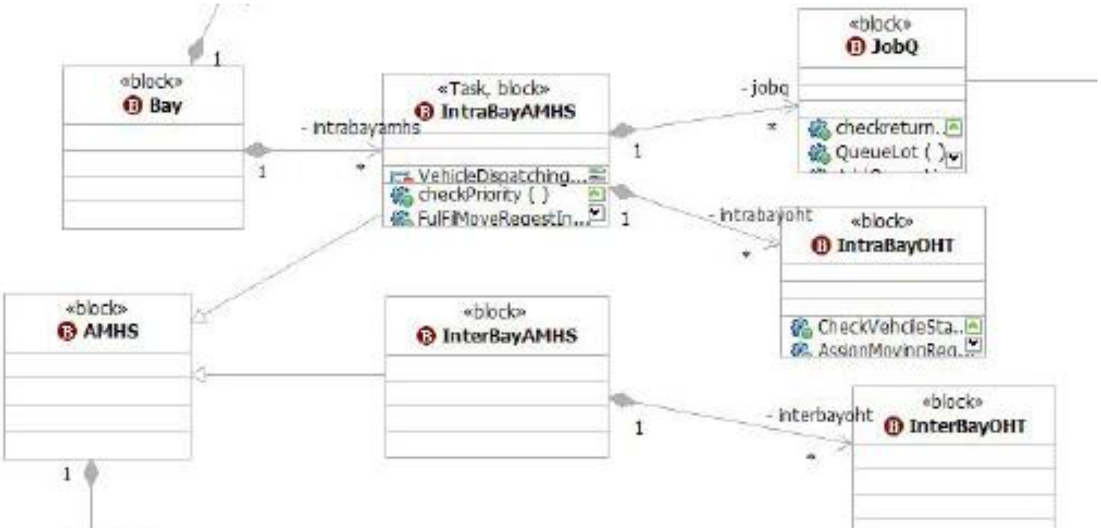


Figure 5. BDD for Bays

Logical structure elements

There are logical as well as physical components of fab structure. In particular the structure of product flow through the fab is dictated by the process plans of the products being produced. Thus every lot in the fab has an associated process plan. The process plan consists of a sequence of operations, and each operation requires a particular tool family, or perhaps even a particular tool. For example, if a lot has the first imaging step performed on a particular lithography tool, then it may be necessary for subsequent imaging operations also to be performed on that same tool. Thus, there is a generic process plan for a class of jobs corresponding to a single product, and an *instance* of the process plan associated with each lot.

Figure 6 illustrates the block portion of the block definition diagram corresponding to these logical elements of fab structure. According to Figure 6, each product has associated with it one or more lots, and a lot contains a set of one or more wafers. A process plan may apply to more than one product, e.g., by using alternative subroutes. A process plan contains one or more recipes, and each recipe contains one or more process parameters. A recipe is associated with a tool group, which is the type of tool for which the recipe is valid, and a route detail, which contains information about the operation as-performed. The route details are collected into a ProcessRoute, which is a specialization of the process plan containing the as-produced information for a particular lot.

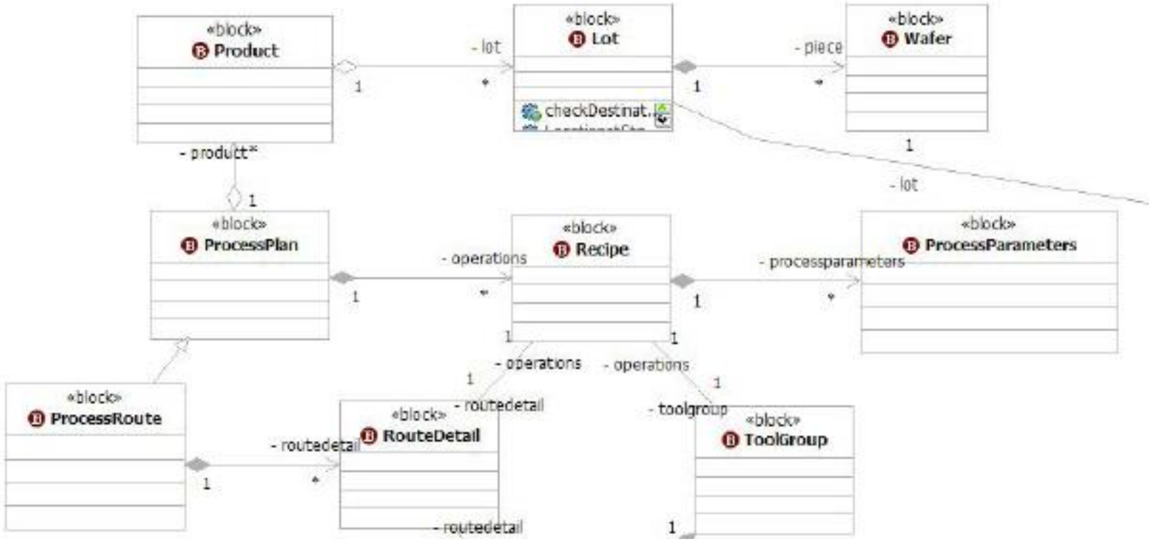


Figure 6. Operations and Process Plans

Integrated Physical Model

The integration of figures 3 through 6 provides a complete conceptual model of the fab at the level of material flow and the systems that perform material flow activities. Note, however, that this is a model of the elements of a fab, and not a model of a specific fab. The fab designer or design team, in order to specify a particular fab, would have to provide the relevant information for all instances of the elements identified. For example, which arcs of the AMHS network are connected, and the location of all the interface and transport points must be specified in order to determine the distance between two particular tools, and the travel time for an OHT to move a lot between them. Likewise, every tool would have to be identified, every product identified and given a specific process plan, etc. We will discuss in section 5 how the SysML models actually support this specification.

Fab Behavior Modeling

From a facility logistics perspective, behavior can be defined as how the fab performs in converting the input of lots (lots released for production) into the output of lots (completed wafers out). Everything that is “interesting” from a facility logistics perspective, is captured in the details of material flow related events, such as the release of a lot, the dispatch of a lot or vehicle, the assignment of a lot to a tool, the loading of a lot on a tool, the completion of a lot on a tool, or the unloading of a lot from a tool. From these time-stamped events, we can compute the time associated with the completion of every step of a lot through its lifecycle in the fab, and we can also compute performance measures for tools, vehicles, and stockers.

In other words, what interests us about fab behavior is the material flow related events. What we need is a comprehensive model that identifies these events, and explains the relationships between them. Our approach to developing such a model is to view each of the physical elements of the fab as a *state machine* with a well-defined set of states (from a material flow perspective). The challenge then is to describe the processes which lead to state-change events, and how the instances of state machines interact with each other.

State Machine Paradigm

Each of the structural elements of the fab, as described in section 4.1, is viewed as a state machine. Figure 7 is an illustration of a state diagram for a generic vehicle, which has three states: "empty assigned," "loaded assigned," and "empty unassigned." An empty but assigned vehicle is deadheading toward the load port where it will pick up it's assigned load. A loaded vehicle is moving toward the destination for the load it is carrying. An empty but unassigned vehicle is idle. Note that idle vehicles may be deadheaded to some location, simply to prevent them from blocking assigned vehicles.

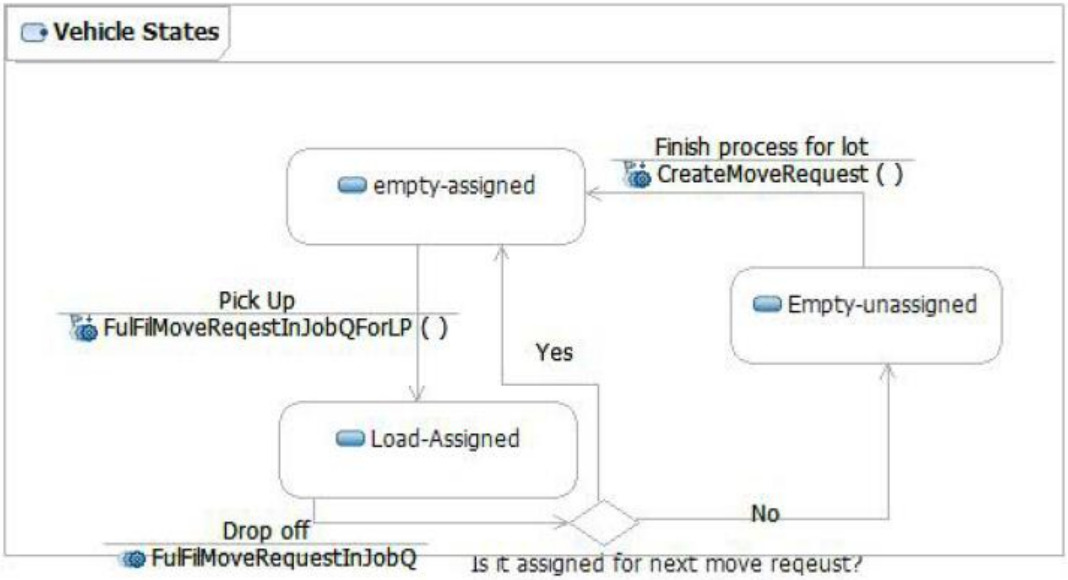


Figure 7. Example Statemachine Diagram for Vehicle

In figure 7, there are two kinds of notations on the arcs representing state transitions. The first is a brief description of what happens to cause a vehicle state change; for example, picking up a load changes the vehicle state from empty assigned to loaded assigned. The second notation, highlighted by the gear icon, identifies a specific type of interaction between the vehicle and some other class in the wafer fab model.

State Machine Interactions

The processes performed by fab structural components (operations on tools, movement by vehicles, etc) may change their own states, or may change the states of other structural components. For example, when a vehicle deposits a FOUP in a load port, it changes state from loaded to empty. At the same time, the port changes state from empty to loaded. The sequence diagram in SysML provides a clear, unambiguous way to represent the interactions, and in particular, to capture sequences of interactions.

Figure 8 illustrates a sequence diagram for the interaction between a port, an intrabay AMHS, the job queue in a bay, and intrabay vehicles. When the port becomes empty, a “create move request” causes the intrabay AMHS to search for an available job to dispatch to the now-available tool port. The AMHS does this by querying the job queue. If the job queue is empty, nothing happens, but if there is a job available, then the AMHS checks its internal status data to determine if a vehicle also is available. If there is no available vehicle, then nothing happens, but if a vehicle is available, the AMHS checks to find the highest priority vehicle, and the highest priority lot, and dispatches the vehicle to move the lot to the available tool.

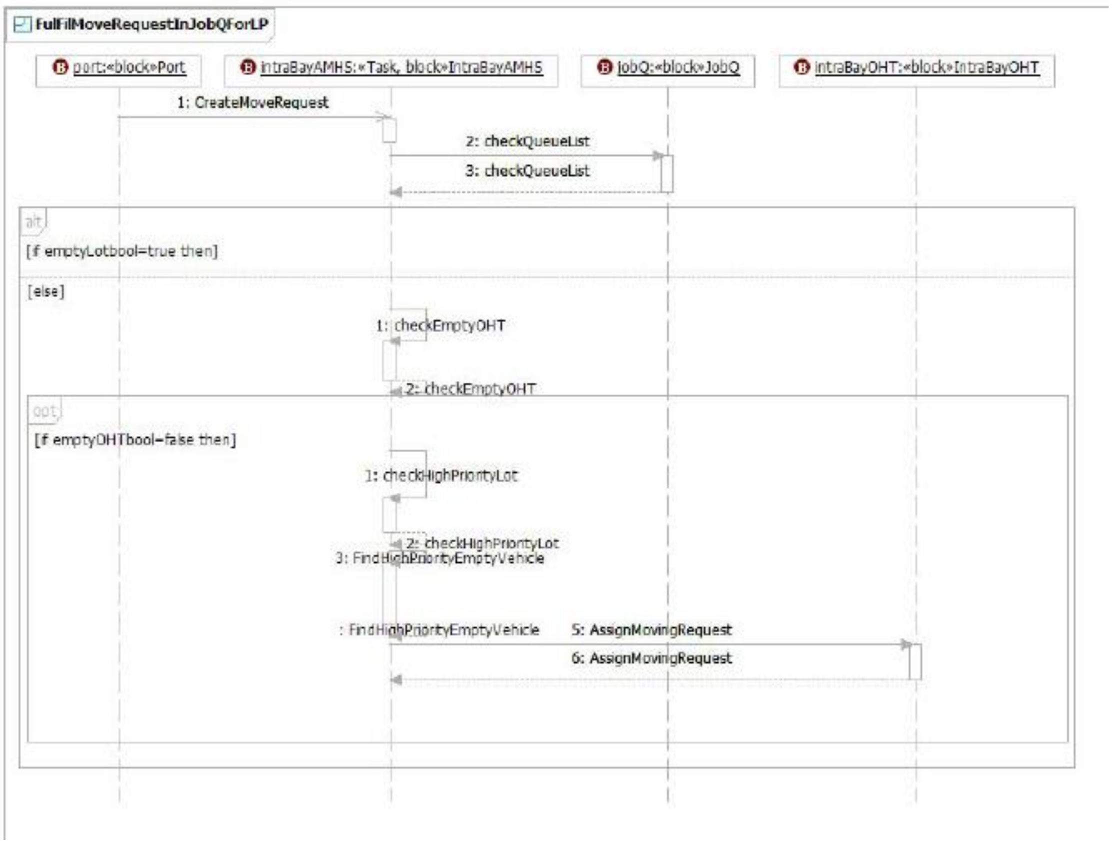


Figure 8. Illustration of Sequence Diagram

Figure 8 illustrates one obvious point, and one not-so-obvious point. First, the sequence diagram syntax enables the description of if-then-else and similar logical structures. Second, the operations identified in the sequence diagram are themselves blocks in the SysML model, with their own properties. For example, the operation "checkQueueList" may represent a particular algorithm to be used to select a candidate lot from the job queue if one exists. Thus, no matter how complex the logic of the processes to be applied at material flow induced state changes, these generic components of behavior can be identified, codified, and most importantly, saved in a library.

SysML Libraries

SysML allows the generic components of the fab structure and behavior to be defined and captured in a library for use in constructing specific fab models. The structure library can contain generic models of all the fab structural components, with generic properties. Similarly, the behavior library can contain blocks representing the generic behavioral actions, such as first-in-first-out dispatching, or priority rule dispatching, and the user can combine these genericl components as needed to represent the desired behavior.

5. Creating and Using SysML Models

SysML provides facilities for creating packages of model elements, and it is object-oriented so hierarchical models could be constructed. Nevertheless, a SysML model with over a thousand tools, and all the guidepath segments would be a very large and somewhat unwieldy model. Furthermore, working directly with such a large SysML model is contrary to the vision we've described.

Rather, we propose the application framework illustrated in Figure 9. In this framework, the domain experts (fab designers) work together with the modeling and analysis experts to create SysML libraries of domain elements. These SysML libraries are translated into the forms necessary to become part of typical fab information authoring tools. For example, the library of fab physical elements might be translated into an AutoCAD™ library for use in FactoryCAD™ to author the fab configuration description. The SysML libraries also can be used to specify schema for databases to capture information about processing operations, process plans, etc. The only need for fab designers to work directly with SysML would be in specifying new sequence diagrams, and even these new sequence diagrams would use library objects for the interactions between entities.

We have demonstrated this library concept for a variety of databases and for FactorCAD. See, e.g., McGinnis, *et al* (2006), Huang, *et al*, (2007), and Kwon and McGinnis (2007).

Thus, using familiar authoring tools, the fab designer creates, and edits a description of the fab being designed. While the information created by the designer conforms to the SysML model, it is not represented as a SysML model. In fact, a large fraction of the design is described in standard database and CAD files. We refer to this as the "instance descriptive model" as it is descriptive rather than analytic, and it is an instance of a fab design specification, rather than a generic (or meta-) model.

The key to *on-demand* analysis of a fab design is the ability to generate an appropriate corresponding instance analysis model directly from the instance descriptive model. The approach for doing this also is described in Figure 9. The off-line activity that creates the SysML domain model and the associated descriptive model libraries also creates analytic model libraries and a model translator. The instance descriptive model is converted by the translator, using the analytic model libraries into a specific instance analytic model, for a specific COTS solver.

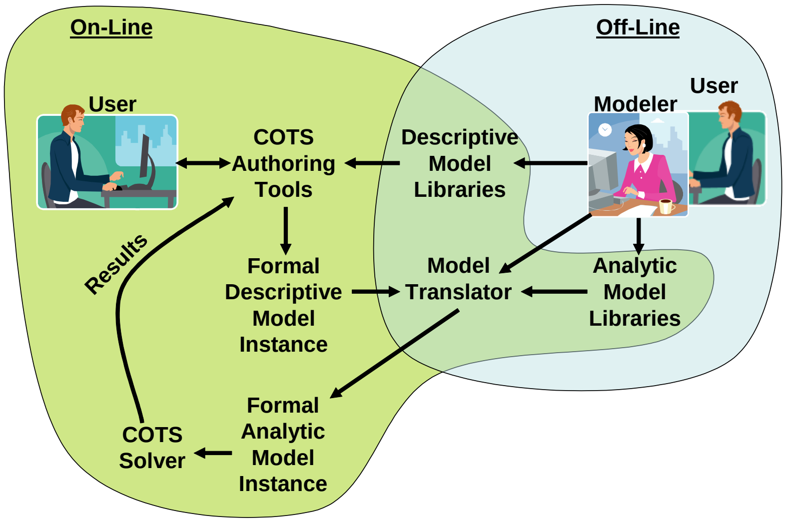


Figure 9. Application Framework

We have demonstrated this approach to *on-demand* analysis for simulation, with eM-Plant ™ as the target COTS simulation tool, see Kwon and McGinnis (2007) and Huang, *et al* (2007).

The framework proposed in Figure 9 represents a radical departure from common practice in the use of simulation in facility logistics. However, this framework is completely consistent with the use of simulation in electrical or mechanical CAD, where the role of the model/analysis expert is an off-line role, and deficiencies in the formal analytic model instance are corrected by improving the descriptive model libraries, the analytic model libraries and/or the model translator.

In addition, this approach clearly is domain-centric, i.e., focused on a specific facility logistics domain, such as wafer fab, or warehouse. We make no claim that this approach could be successful if applied to the generic facility logistics problem. However, we also would not rule out the possibility that applying the framework to specific facility domains would lead to a more generic understanding of facility logistics system design, and a more generic suite of facility logistics system design tools.

6. Implications and Challenges

After two years of experimentation with SysML, we believe it to be a sufficiently powerful and expressive systems modeling language to be effective in describing and analyzing facility logistics design problems. In the two years of experimentation with the language, we have evolved the application strategy portrayed in Figure 9, and have demonstrated every major element of that strategy. The implication we draw from this experience and these results is: ***“There is no fundamental impediment to the creation of powerful facility logistics CAD/CAE tools, other than the time and budget to productize proven research results, and extend those results to additional domains.”*** Furthermore, because our research efforts, at every opportunity, have employed COTS tools, we believe the required time and budget to be quite reasonable.

While our work has demonstrated the potential of our proposed approach, a number of important research questions remain open:

- The application framework allows a number of users to engage a number of COTS authoring tools, but is silent on the management of the associated workflow; what modeling and analysis support will be needed to enable a seamless, efficient, and correct design workflow?
- The application framework focuses on the model libraries, but the underlying domain model is critical to their creation; what are the most effective modeling concepts, model organizations, and ontologies to support the framework?
- The application framework has been demonstrated for high fidelity simulation models; can the framework be shown to support facility logistics systems design from initial conceptual development all the way through to detailed engineering design?

Clearly, there is opportunity for many researchers to address important questions leading to a more thoroughly grounded engineering approach to facility logistics system design.

7. Acknowledgements

This work has benefited enormously from the interactions with Dr. Russell Peak of the Manufacturing Research Center and Dr. Chris Paredis in Mechanical Engineering, and their students. Together, we all have been exploring the use of SysML and various SysML software tools. In addition, the work specifically addressing facility logistics has benefited from the legacy of related work in Georgia Tech, starting with the Material Handling Research Center, and extending through the Keck Virtual Factory Lab.

8. References

Huang, E., Ramamurthy, R. and McGinnis, L.F., System and simulation modeling using SysML, Winter Simulation Conference Proceedings, Pages: 796-803 ,2007

Gu, J., Goetschalckx, M. and McGinnis, L.F., Research on warehouse operation: A comprehensive review, European Journal of Operational Research ,Vol 177(1), Page(s) 1-21

Kwon, Kysang; McGinnis, L.F., SysML-based Simulation Framework for Semiconductor Manufacturing ,Automation Science and Engineering, 2007. CASE 2007. IEEE International Conference on, 22-25 Sept. 2007 Page(s):1075 – 1080

McGinnis, L.F., Huang, E., and Wu, K. , Systems engineering and design of high-tech factories, Winter Simulation Conference Proceedings, Pages: 1880 - 1886 ,2006

OMG , OMG Systems Modeling Language (OMG SysML™), V1.0 Specification, Available: <http://www.omg.org/docs/formal/07-09-01.pdf > or < http://www.omgsysml.org/#Specification>, 2007.