A COMPONENT-BASED APPROACH FOR MANUFACTURING SIMULATION

Frank Riddick
Deogratis Kibira
Y. Tina Lee

Manufacturing Systems Integration Division
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20899, USA
Frank.Riddick@nist.gov

Stephen Balakirsky

Intelligent Systems Division
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20899, USA
Stephen.Balakirsky@nist.gov

Keywords: Manufacturing, component-based architecture, reference architecture

Abstract
Manufacturing systems can be very complex and are often costly to develop and operate. Simulation technology has been shown to be an effective tool for optimizing manufacturing system design, operations, and maintenance procedures. However, each manufacturing simulation is usually developed to address a specific set of industrial issues, and may only apply to a small portion of a complex manufacturing system. To enable manufacturers to more easily use simulation technology to solve complex manufacturing issues, this paper defines a reference architecture for component-based simulation (RACS). With the architecture, complex manufacturing systems are functionally partitioned into smaller interacting subsystems, and simulations of those subsystems are combined to form a federated simulation of the overall manufacturing system. This enables the simultaneous analysis of different aspects of each of the simulated subsystems and the overall manufacturing system.

1 INTRODUCTION

To confront the challenges of today's global, ultra-competitive marketplace, many manufacturers have embraced the concepts of agile manufacturing [1]. Agile manufacturing is a philosophy or approach where companies seek to organize and carry out their operations in a manner in which they are able to cope with and possibly benefit from today's complex, every changing, global manufacturing environment. Much has been written about agile manufacturing, and although there is no consensus on what technologies are necessary to implement an agile manufacturing program, tools that support rapid prototyping and integrated product/process development have repeatedly been mentioned [2-3].

Simulation technologies have been employed to solve problems in manufacturing and to enable manufacturers to be more agile. Many simulation technologies enable the creation of virtual representations of factory facilities, machines, material handling systems, support applications, robots, and production processes. Once created, virtual factory representations can be used to analyze different aspects of current factory operations, and to plan for and analyze prospective changes to the factory environment. This virtual product and process prototyping capability enables the analysis of product and production system changes without the need for expensive full-up product mockups or physical changes to the production facilities.

A problem impeding the ability of manufacturers to use simulation technologies is that simulation applications exhibit a low level of interoperability, both between simulation applications and with other manufacturing applications. Manufacturing simulation applications are usually monolithic and provide few avenues for integrating with other applications. Often when outward facing interfaces for integration are provided, they are undocumented and/or proprietary. In addition, the underlying technologies used to create simulation applications are often different. Simulation applications might be based on the discrete-event simulation approach, be made up of a collection of interacting software agents, operate based on the effect of the physical properties of the simulated parts and machines with the simulated environment, or be constructed based on a heterogeneous collection of these and other approaches. Even though manufacturing enterprises can save time and money by analyzing simulated representations of the production facilities and processes, interoperability issues make creating and using simulations costly and time consuming. This one issue is a serious obstacle that manufacturers must overcome in their quest to become more agile.

To enable manufacturers to more easily use simulation technology to solve complex manufacturing issues and to be more agile, in this paper a reference architecture for component-based simulation (RACS) is proposed. The architecture is based on prior research at the National

Institute of Standards and Technology (NIST) in simulation integration [4]. In this component-based architecture, the aspects of a manufacturing system that are to be analyzed are functionally partitioned into subsystems. The subsystems together define the operation of the complete system by exchanging messages that are a part of a shared message protocol. Software components that are simulations of each of the subsystems can then be created for each of the subsystems. By using an integrating infrastructure to enable message exchange and time coordination, the individual components can be made to function together as a combined "federated simulation" that simulates the operation of the complete manufacturing system.

The rest of the paper is as follows. Section 2 provides an overview of the architecture. In section 3, a case study in which a federated simulation will be created is presented. Descriptions of the manufacturing system that is to be simulated, how it is partitioned into subsystems, the function of each subsystem, alternatives for the components that will simulate each subsystem, and the requirements for the integration infrastructure are also presented. The paper concludes with section 4, where a summary and description of future work is presented.

2 A REFERENCE ARCHITECTURE FOR COMPONENT-BASED SIMULATION

2.1 Overview

The reference architecture for component-based simulation (RACS) provides a blueprint for creating software applications that enable the simultaneous analysis of multiple aspects of a complex manufacturing system. The applications created based on the architecture are made up of several individual simulation components that each individually simulates some aspect of the manufacturing system. The simulation components coordinate their advancement through time and exchange information as messages so that their aggregate behavior simulates the behavior of the complete manufacturing system being studied. With this approach, each simulation can be constructed to perform a more detailed analysis of some aspect of the manufacturing system while simultaneously contributing to and supporting the simulation and analysis of the behavior of the complex manufacturing system as a whole. Applications constructed as conglomerations of interacting simulation components are frequently referred to as federated simulations. Figure 1 depicts the architecture and many kinds of simulation components that might be constructed to run on it.

Applying the architecture to create a federated simulation application that is able to analyze a complex manufacturing problem involves several steps:

1. Manufacturing System Definition and Analysis

   In this step, the overall function of the manufacturing system to be studied is defined. The form and level of detail of the definition need only be specific enough to support the definition of the simulation components that will be used to create the federated simulation. Also during this step, any data that must be produced by an individual simulation component because it is required to support an analysis of the manufacturing system as a whole should be specified.

2. Manufacturing Subsystem Definition

   This step involves functionally partitioning the manufacturing system into subsystems. The data and behavior associated with each subsystem should be identified. Information that needs to be shared between the subsystems should be minimized. Each subsystem should be able to operate semi-independently, with as few information and functional dependencies on the other subsystems as possible. Also, when considering how best to partition the system into subsystems, the kinds of analysis that a simulation component might perform for each subsystem should be taken into account.

3. Message Protocol Definition

   In this step, the information that needs to be exchanged between subsystems so that they can replicate the required behavior of the overall system is defined. This is accomplished by defining messages that can be sent from each subsystem to the other subsystems, the content of those messages, and the conditions that stimulate each message to be sent. The message protocol and subsystem definitions from step 2, when taken together, form a detailed definition of the overall system behavior described in step 1.

4. Simulation Integration Infrastructure Design and Implementation

   At this point, the infrastructure that will support the federated simulation of the manufacturing system must be defined. The infrastructure must be able to allow the simulation components to exchange messages and to coordinate their advancement through time during simulation execution. There are a multitude of approaches that can be taken in designing the infrastructure, including: designing and creating the infrastructure from scratch using basic computer language tools; modifying general purpose distributed computing middleware products; and, using

middleware specifically designed for creating federated simulations. A detailed discussion of some of these options is provided in section 3.3.

An optional component of the simulation integration infrastructure is the Federated Simulation Management component. Depending on how the infrastructure is defined, there may be a need for some common low-level support tasks to be performed that are outside of the responsibilities of the simulation components that will represent the manufacturing subsystems in the federated simulation. The tasks most commonly implemented are simulation execution start/pause/stop and data logging. Sometimes, one of the simulation components can be implemented to cover the required services.

5. Simulation Component Design and Implementation

For each subsystem, create a new or adapt an existing simulation that: (1) implements the functional responsibilities defined for that subsystem; and, (2) performs an analysis of some aspect of the subsystem. The functional responsibilities involve mainly implementing the message protocol and coordinating time advance with the other simulation components through the simulation integration infrastructure.

After all of the components and the infrastructure have been designed and implemented, they can be run together to create a federated simulation that enables analysis of the complex manufacturing system that was described in step 1 of the development process.

There are several advantages for constructing applications in this way.

Divide and conquer

Analysis of a complex manufacturing system can be accomplished by partitioning the problem into implementable components.

Scaling

Since some infrastructures support the distribution of the simulation components over different computers, the analysis of larger problems can be done.

Conceptual analysis of the manufacturing system

The initial analysis of the manufacturing system and its functional partitioning into subsystems often provides insight into the behavior of the system before any simulation components are created.

Simultaneous analysis

Since each simulation component is created to support the

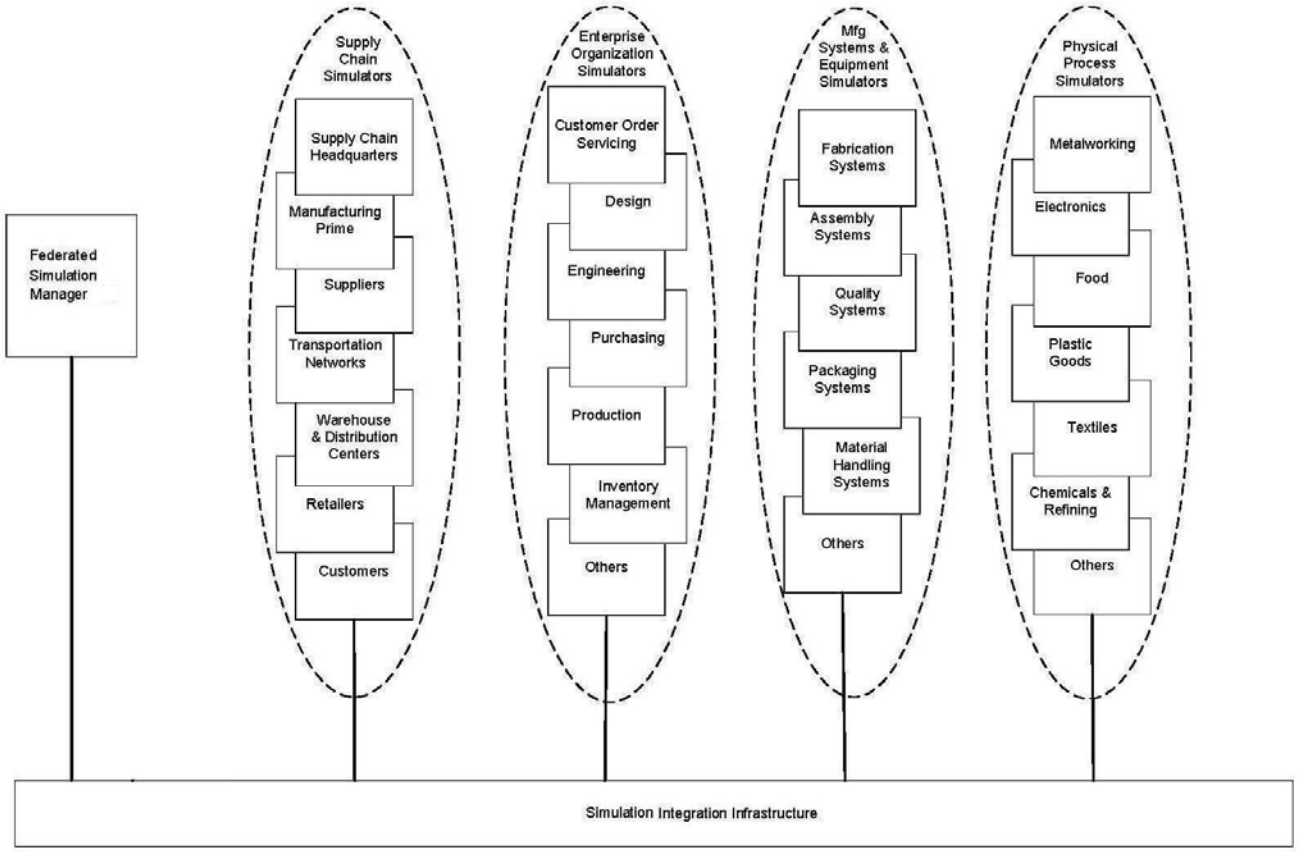

Figure 1 - The Reference Architecture for Component-Based Simulation (RACS)

overall system analysis and the component can be created to perform some analysis on the subsystem it represents, multiple analyses of the manufacturing system can be done simultaneously.

Simulation component reuse

While in aggregate the simulation components implement the overall behavior of the main manufacturing system, the internal design and behavior of the components can be very different. This allows multiple components to be designed that adhere to the same message protocol, but do different kinds of analyses for the subsystem being simulated.

2.2 Elements of the architecture

While the architecture facilitates the definition on many different federated simulation applications, it has only three architectural elements: simulation components, the simulation integration infrastructure, and the federated simulation manager.

Simulation Components

A simulation component is a software application that simulates the behavior of some subset of a manufacturing system or subsystem. It may be constructed from a number of technologies, including general computer languages (e.g., C#, C++, and Java), commercial discrete-event simulation creation packages, software agent systems, and scientific computing tools (e.g., MatLab and Modelica based tools).

The kinds of analysis that a component may do is limited only by the skill of the implementers. The only requirement for a simulation component to be a part of a federated simulation is that it implements the messages defined in the message protocol and it coordinates with the other simulation components using the simulation integration infrastructure.

Simulation Integration Infrastructure

The simulation integration infrastructure is a software application that enables simulation components to exchange information as messages and to coordinate the advancement of time. The infrastructure may be created from a number of technologies, some providing only minimal communication services and others providing a rich set of implementation possibilities. Often this architectural element is created based on existing distributed computing middleware (e.g., Sun Jini [5] or the Common Object Request Broker Architecture (CORBA) [6]) or middleware designed to support federated simulation development (e.g., High Level Architecture (HLA) [7] or the Synchronous Parallel Environment for Emulation and Discrete-Event Simulation (SPEEDES) [8]).

Federated Simulation Manager

The Federated Simulation Manager is an optional component that can be used for basic support services during federated simulation execution. These services usually involve initial simulation synchronization, simulation starting and stopping, error collection, and data logging. Depending on the simulation integration infrastructure chosen, some of these services may already be provided or may be implemented as a part of one of the simulation components.

3 APPLYING THE REFERENCE ARCHITECTURE

In this section, an example describing how to apply the first steps of the reference architecture is presented. A complex manufacturing system is described, a functional partitioning of the system into three subsystems is presented, and a message protocol is described. For each subsystem, descriptions for one or more simulation components that could represent that subcomponent's functionality in a federated simulation of the manufacturing system are presented. In the descriptions of the simulation components, alternatives for how they might be constructed and what kinds of analyses they could perform on the simulated manufacturing subsystem are presented. Alternatives for how the simulation integration infrastructure might be designed and implemented are also presented. The actual implementation of the components and infrastructure is not presented, but is expected to be the object of future work.

3.1 Manufacturing System Conceptual Overview

The manufacturing system that is to be modeled is a discrete parts production environment. The system should be modeled in such a way that studies can be undertaken that examine the effect on system performance of changes to product mix, automated guided vehicle (AGV) delivery schedules, and consumable part palletizing.

To avoid the problems that would be caused by attempting to model this system as one complex monolithic entity, the functionality of the system will be partitioned into three functional subsystems. The Production subsystem will model the workstations that perform the manufacturing operations that transform workpieces from one state to another until they are finished products. Material handling to transport workpieces will also be modeled by the Production subsystem. The Inventory Management subsystem will model the process by which requests for consumable parts by workstations in the Production subsystem get turned into organized pallets of parts ready for delivery to the requesting workstations. The AGV Management subsystem will model AGVs, their dispatching

to pick up and deliver parts, and their movement through the production environment.

In Figure 2 a Unified Modeling Language (UML) interaction diagram is presented that shows each subsystem of the manufacturing system being studied and the interactions between those subsystems. In the sections below, a description of the behavior of each subsystem and its interactions with other subsystems is presented.

3.1.1 Production subsystem

The Production subsystem is a representation of a discrete parts production facility that allows the exploration of manufacturing issues related to inventory management and workstation material replenishment. This system is made up of a number of workstations connected by fixed material handling systems. The material handling systems are used to transfer workpieces from workstation to workstation. Workpieces are partially finished parts and subcomponents that will eventually be transformed into the finished products.

The facility can produce a small number of different finished products, and many of the products share common subcomponents. Each of the products that can be produced by the facility has a process plan that defines the sequence of steps necessary to produce that product. Each step in a process plan defines:

- The type and amount of each workpiece involved
- The consumable materials/subcomponents introduced into the manufacturing process

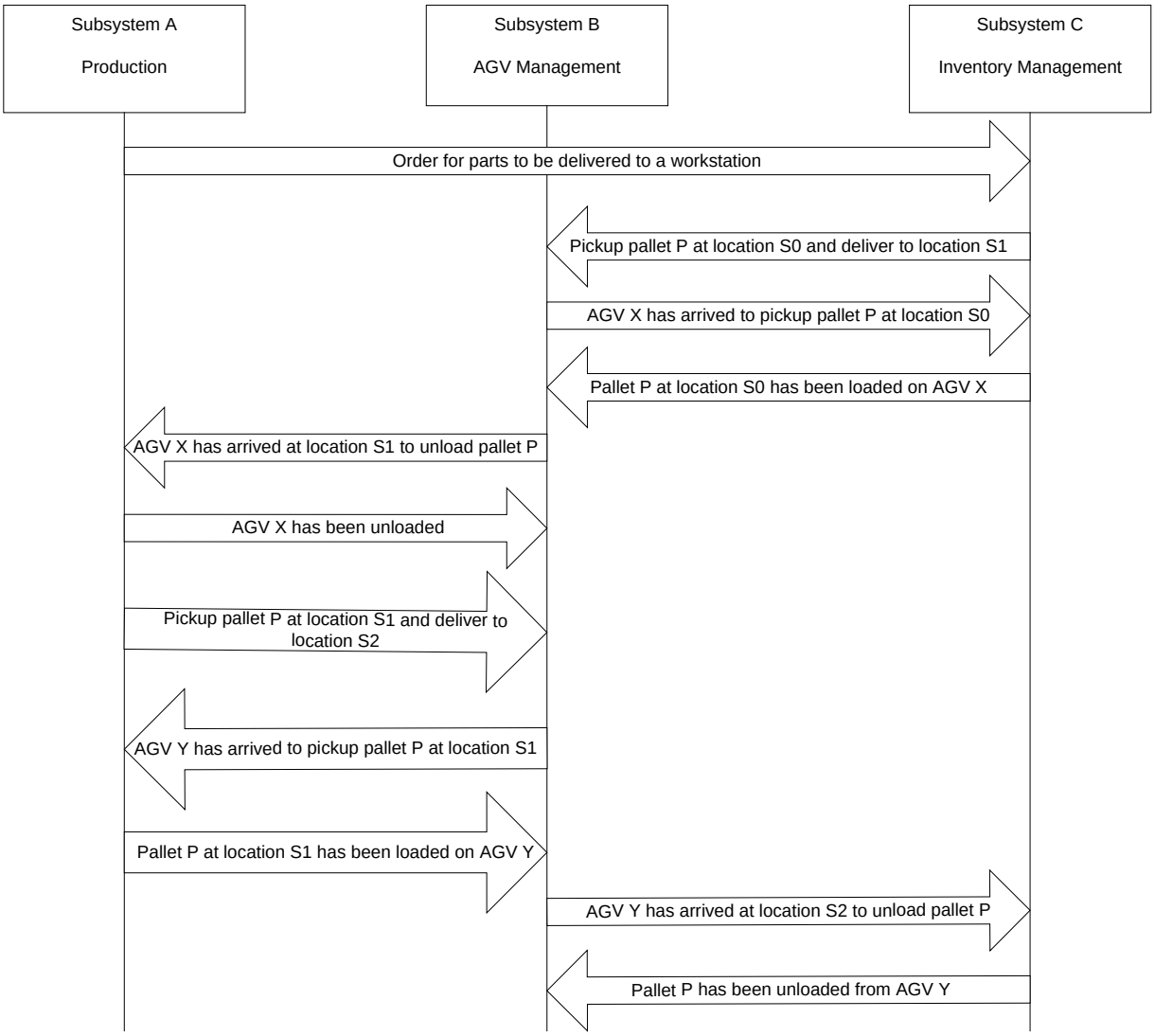

Figure 2 - Message exchange between subsystems for part pickup and delivery

- The type and amount of workpieces produced as output
- The production operation that will be executed
- The workstation on which this step will be executed

Each workstation is configured with the process plan information for each operation that can be performed at that workstation. This information allows each workstation to automatically choose and execute the appropriate processing step when presented with workpieces by the material handling system. This approach allows each step in the product's process plan to be executed in sequence to produce the final product without requiring direction from a central controller.

While workpiece movement is handled by the material handling system, the management of consumable parts at each workstation is not. At each workstation, storage bins are provided for each consumable part that can be involved in a production operation at that workstation. A reorder level amount is set for each consumable. When during the course of production the amount of a consumable goes below the reorder level for that consumable, a request for replenishment of that consumable is sent to the Inventory Management subsystem. The Production subsystem will continue performing its production activities as long as it has enough consumables available.

At some point after the request for replenishment, an AGV will arrive with the consumables requested by the workstation. The Production subsystem will coordinate with the AGV to unload the consumables, store them in the appropriate bins, update the amount on hand for each consumable, and indicate to the AGV when it is finished unloading.

3.1.2 AGV Management

The AGV Management subsystem represents both the automated guided vehicles (AGVs) that pick up parts from inventory and deliver them to production workstations, and the management application that monitors and directs the AGVs. The management application:

- Accepts requests from the Inventory Management subsystem to pick up parts that are loaded onto pallets for delivery to production workstations
- Dispatches AGVs to pick up pallets and deliver them to specific production workstation locations
- Monitors the performance of each AGV on its assigned task
- Coordinates with the Inventory Management subsystem and the Production subsystem to accomplish AGV loading and unloading

The AGV Management subsystem may employ different AGV dispatching strategies depending on the number of AGVs under its control, whether the AGVs follow fixed or ad-hoc delivery routes, the overall production goals, and other factors.

3.1.3 Inventory Management

The Inventory Management subsystem provides two main functions for the manufacturing facility. It monitors and maintains appropriate levels of inventory items that are the consumable parts needed for production. In addition, when requested it retrieves from storage the inventory items needed to keep production operations running and packages them for efficient delivery to the production workstations that need the items. Items for delivery are packaged on equally sized pallets, and the Inventory Management subsystem coordinates with the AGV Management subsystem for pickup and delivery of the pallets and for return of empty pallets.

Of particular importance to Inventory Management subsystem performance is how inventory items are packaged for delivery, referred to as the palletizing process. An effective palletizing process can greatly affect overall production performance by minimizing production delays due to late delivery of parts. Also, wear and tear on transportation equipment can be minimized by not creating loaded pallets with too much weight and by avoiding unnecessary delivery trips caused by poor packing of items.

3.2 Simulation Component Design Alternatives

In the previous sections, high-level descriptions of three key subsystems of a complex manufacturing system, and the interactions and interrelationships of those subsystems, were presented. By partitioning the functionality of the manufacturing system in this way, simulation components dedicated to the examination of the detailed behavior of each subsystem can be more easily constructed. Furthermore, with the addition of an integrating infrastructure, the individual simulation components can be made to operate together as a federated simulation.

To enable the component simulations to be able to operate as a federated simulation, each simulation must implement and adhere to a specific message protocol. The message protocol defines the content of and conditions under which messages are sent between components. The requirements for the message protocol that will be used in this study were described in Section 3.1 and its subsections. Later, in section 3.3, design alternatives for implementing an infrastructure to support the message protocol are presented.

In sections 3.2.1 to 3.2.4, descriptions of designs for the components that will simulate the functionality of each of the manufacturing subsystems are presented. For the AGV Management subsystem, multiple descriptions for

simulation components that might implement the subsystem's functionality are given.

A key feature of the component-based simulation approach is that as long as a simulation component adheres to the agreed upon interaction protocol, how that component is implemented internally should not fundamentally change the overall functional behavior of a federated simulation involving that component. Items such as how long it takes to run and what hardware and support software are needed may be affected, but any data produced by or the types of analysis that can be performed by the federated simulation should not change. This feature of the component based simulation approach facilitates the development of simulation components that are created using vastly different simulation technologies and methodologies [9]. When multiple implementations of simulation components of a manufacturing subsystem are available, they can be used and reused to create different federated simulations focused on analyzing different aspects of the manufacturing system being simulated.

3.2.1 Production – Discrete Event Simulation Component

The production simulation component is a visual model of resources, coordination, and control of the activities that take place on the manufacturing floor. The purpose of such a model is to enable the analyst to investigate and optimize shop floor operations, and send and receive messages to the other simulation components in the federation for smooth operation of the simulated shop. The simulation application to be used should model the Production subsystem elements, i.e., automated machine tools, inspection equipment, material handling systems, storage buffers, and transfer robots. It should represent entities, i.e., the parts that get assembled to produce the product, control elements as well as tools such as, cutters, hand tools, jigs and fixtures. In addition, the model should be able to maintain a list of AGV calls, show status of resources, and have integration mechanisms with other simulations, processes, and databases. There are a number commercial off-the-shelf (COTS) discrete-event simulation applications for manufacturing systems from which a suitable candidate could be selected.

3.2.2 AGV Management – Physics-based Component

A physics-based AGV Management component will combine a high-level multi-vehicle control system with a physics-based vehicle simulator [10]. The high-level controller will not be simulated. It will be a real-time commercial or research grade multi-vehicle controller that will receive requests for goods and services (e.g., delivery of pallets or removal of empties and defects) and will determine vehicle loading, routes and schedules for its fleet of vehicles. It must be able to handle various priorities of requests, traffic management, and a dynamic factory floor that has a constantly changing topology. This controller will send commands and receive status from a fleet of AGVs that are physics-based simulated entities over the same channels and with the same format as it would use for real vehicles.

The simulated AGVs will operate in real-time and will travel over the commanded routes provided by the high-level controller. The models will include sensor and mobility platform models so as to realistically simulate low-level vehicle performance while traversing the commanded routes. The simulation environment will provide dynamics, which the vehicles must be capable of responding to (e.g., moving avatars or non-robotic vehicles).

3.2.3 AGV Management – Process-oriented Component

A process-oriented AGV Management component might be developed when the kinds of analyses to be done in the federated simulation do not depend on virtual representations of the physical characteristics of the AGVs. In this approach, a dispatching application uses a set of rules to determine which AGVs are assigned to pick up and deliver pallets of parts and stochastic variables are used to determine simulated AGV delivery times. In such a scenario, the component could be designed to focus on issues such as determining the optimum number of AGVs required to handle the expected delivery workload, or how should the dispatching rules be changed if several AGVs need to be taken out of service for maintenance. A component such as this would be unable to explore some issues, such as determining optimum AGV path finding and collision avoidance strategies.

3.2.4 Inventory Management and Palletization – Physics-based Component

A physics-based inventory management and palletizing component will be a combination of inventory control, pallet planning, and pallet building subsystems. The inventory control and pallet planning subsystems will utilize commercial or research grade systems that are capable of receiving and fulfilling orders. Orders will be received from individual machine stations in a standardized XML format. The inventory control system will determine the part's availability and create an XML formatted packing list that will be sent to the pallet planning subsystem. The pallet planning subsystem will then create plans for one or more mixed pallets of goods that include pallet build schematics and ordering information for parts to be placed on the

conveyor systems. This information will be utilized by the physics-based robotic system to simulate the construction of the actual pallets for transport by the AGVs. By simulating the construction of the pallets, the stability of the pallets may be evaluated and the overall quality of the packaging solution may be evaluated.

3.3 Infrastructure Design Alternatives

Determining the best approach for designing and implementing the integrating infrastructure can be a complicated undertaking. The basic requirements are rather straightforward: (1) providing a means for simulation components to exchange data, and; (2) providing a means for the simulation components to coordinate the advancement of time with each other. These capabilities must be provided for the component simulations to be combined to form a federated simulation.

To enable the system to better support the agile manufacturing paradigm, several additional requirements should be met, including: (3) allowing different collections of simulation components that support the same message protocol to be a part of a federated simulation; (4) enabling the implementation of simulations best-fitted to analyze different aspects of their associated subsystems or of the overall system, and; (5) allowing simulation components to be created with different technologies. Although somewhat high level, this list of requirements defines an achievable target for infrastructure functionality.

Given the list of desired functionality for an infrastructure, what technology or technologies can be used for infrastructure implementation? One approach is to use middleware specifically designed for distributed simulation creation, such as the HLA Run-Time Infrastructure (RTI). Another approach is to design the infrastructure from the bottom up using general-purpose computer languages, and using sockets and pipes for communication. Alternatively, implementing the infrastructure can be accomplished using general-purpose distributed computing technologies, such as the Neutral Message language (NML) [11] or CORBA.

4 SUMMARY

The component-based simulation framework described in this paper fosters agility by enabling the description and study of complex, dynamic manufacturing systems. A scenario was presented that showed how the RACS approach could be used to describe a discrete parts production environment and several of the subsystems that compose this environment. Component simulations of these subsystems can be implemented using different commonly available technologies.

DISCLAIMER

Company names and products may have been identified in the context of this paper. This does not imply a recommendation or endorsement of the software products by the authors or NIST, nor does it imply that such software products are necessarily the best available for the purpose.

5 REFERENCES

[1] L. Goldman, R.L. Nagel and K Preiss, "*Agile Competitors and Virtual Organizations - Strategies for Enriching the Customer*," Van Nostrand Reinhold, New York, NY, 1995.

[2] A. Gunasekaran, "Agile manufacturing: enablers and an implementation framework," international journal of production research, vol. 36, no. 5, 1223 – 1247, 1998.

[3] S. Jain, "Virtual Factory Framework: A Key Enabler for Agile Manufacturing," in Proceedings of 1995 INRIA/IEEE Symposium on Emerging Technologies and Factory Automation, Paris, vol. 1, p. 247-258, 1995.

[4] C. McLean, S. Jain, F. Riddick, and Y. T. Lee. "A Simulation Architecture for Manufacturing Interoperability Testing," in Proceedings of the 2007 Summer Computer Simulation Conference, May 1, 2007.

[5] J. Waldo, "*The Jini Specification*," Addison-Wesley, Boston, MA, 1999.

[6] Object Management Group, "Common Object Request Broker Architecture," Object Management Group, Framingham, MA, 1995.

[7] F. Kuhl , R. Weatherly, and J. Dahmann, "*Creating Computer Simulation Systems: An Introduction to the High Level Architecture*," Prentice Hall PTR, Upper Saddle River, NJ, 1999.

[8] J. Steinman, "SPEEDES: Synchronous Parallel Environment for Emulation and Discrete Event Simulation," in Proceeding of the SCS Western Simulation Multiconference, Anaheim CA, 1991.

[9] A.W. Brown and K.C. Wallnau, "The Current State of CBSE," IEEE Software, September/October 1998.

[10] S. Balakirsky, C. Scrapper, and E. Messina, "Mobility Open Architecture Simulation and Tools Environment," Proceedings of the 2005 Knowledge Intensive Multi-Agent Systems (KIMAS) Conference, Waltham, MA, April 18-21, 2005.

[11] W. Shackleford, F. Proctor, and J. Michaloski, "The Neutral Message Language: A Model and Method for Message passing in Heterogeneous Environments," in Proceedings of the World Automation Conference, 2000.