Proceedings of the 2001 Winter Simulation Conference
B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, eds.

SIMULATION OF SHIPBUILDING OPERATIONS

Charles McLean
Guodong Shao

Manufacturing Simulation and Visualization Group
National Institute of Standards and Technology (NIST)
100 Bureau Drive, MS 8260
Gaithersburg, MD 20899, U.S.A.

ABSTRACT

This paper discusses the objectives and requirements for a shipbuilding simulation. It presents an overview of a generic simulation of shipbuilding operations. The shipbuilding simulation model can be used as a tool to analyze the schedule impact of new workload, evaluate production scenarios, and identify resource problems. The simulation helps identify resource constraints and conflicts between competing jobs. The simulation can be used to show expected results of inserting new technologies or equipment into the shipyard, particularly with respect to operating costs and schedule impact. The use of DOD High Level Architecture (HLA) and Run Time Infrastructure (RTI) as an integration mechanism for distributed simulation is also discussed briefly.

1 INTRODUCTION

Scientists and engineers working on the Manufacturing Simulation and Visualization Program at the National Institute of Standard and Technology (NIST) performed the research presented in this paper. The goal of the Manufacturing Simulation and Visualization Program is to develop data interfaces and test methods for integrating manufacturing simulation and visualization applications to improve the accessibility and interoperability of this technology for U.S. industry.

The National Shipbuilding Research Program (NSRP) in part sponsored the research presented in this paper. The NSRP is a five-year program to achieve significant technology and process improvements in the U.S. shipbuilding industry. NRSP is being carried out as a collaboration among U.S. shipyards, government, industry and academia.

The NSRP has six focus areas, which are listed below:

- Shipyard Production Process Technologies
- Business Process Technologies
- Product Design and Material Technologies
- System Technologies
- Facilities and Tooling
- Crosscut Initiatives.

The Shipyard Production Process Technologies is a major initiative that addresses all production processes used to transform raw material, components, and equipment into completed products (NSRP 2001). The simulation of shipbuilding operations presented in this paper was carried out as part of this initiative. For background on shipbuilding operation, see (Storch 1995).

The objective of the simulation model for shipbuilding operations is to provide data to support shipyard management decisions including:

- Analysis of schedule impact due to additional projects, differing production scenarios, and so forth
- Identification of labor resource conflicts by craft and skill level
- Analysis of cost and effects of exercising overtime, new hire, and/or subcontractor to cover labor shortages
- Analysis of cost tradeoffs of holding over employees without work, versus lay-off, rehires and new hires
- Prediction of optimum staffing levels based sales and labor demand forecasts
- Demonstration of the expected results of inserting new technology or equipment into shipyard, particularly with respect to operating costs and schedule impact
- Visual display of work location, resources, and identification resource constraints and conflicts between competing jobs (including cranes and forklifts)

- Archiving of simulation runs as files to preserve historical data.

2 SHIPBUILDING SIMULATION OVERVIEW

2.1 Functional Requirements

Simulation models can help to identify project and resource management issues that have a major cost impact for manufacturing industry. Project management systems can realistically provide only a static view of project definition, resource, and constraint data. Simulation can be used to model and evaluate complex interactions between overall project workload and available resources. It also can be used to identify resource conflicts, "labor-hiding," and obtain a big picture of overall operations that is not available by other means. Multiple runs of models can provide sensitivity data on the potential impact of random variations in operations.

The next section outlines the simulation system architectures from the views of simulator, control logic, user interface, and internal data management.

2.2 Simulation System Architecture

Simulation system is used to refer to not only the simulation model and the simulation engine, but also the other software applications that are used to generate data for the model, a user interface system, and associated data files. The simulation system is divided into the following component elements:

- Simulator
- Control Logic
- User interface
- Internal Data Management.

These elements are briefly introduced below.

Simulator – The simulation engine used to implement the simulation system in this paper is ProModel 4.22. ProModel is a commercial simulator that can be used for evaluating, planning, or re-designing manufacturing, warehousing, and logistics systems. The simulator allows users to build a graphical representation of an application system and test it in a variety of scenarios to provide better solution for the problems in the organization. The graphical animation and reports are useful tools for visualizing, understanding, and improving the real system. The development simulator includes functions for developing and running models. A separate graphics library is used to generate displays. A Microsoft COM interface is included that allows the simulator to execute external code. The COM interface was used to implement a remote user interface (PRG 1996).

Control Logic – The control logic is the "brain" of the simulation model. It is customized code and data developed to model the production process of the shipyard. The control logic for the simulation model manages the execution of the following modules:

- Graphical user interface
- Data input
- Job and task management
- Production area allocation and space management
- Labor allocation
- Resource allocation
- Calendar and clock
- Report generator
- Remote user interface.

All of the capabilities of these modules are functions built on top of basic ProModel capabilities. More details are provided in Section 2.3.

User Interface – The user interface is the "face" of the simulation model, the animated graphic display allows the user to interactively use the simulation model to evaluate and analyze the production process model of the shipyard. Nine user interface screens were developed as showed in Figure 1 (PUG 1996), including:

- Simulation status
- Space allocation
- Location status
- Production schedule
- Job status
- Task status
- Resource allocation
- Part and inventory status
- Statistical data summary.

These simulation screens can be used to monitor the system remotely by using distributed simulation mechanisms discussed in section 2.5.

Internal Data Management – Four types of data files are currently used by the simulation system:

- Project management data (one for each project)
- Schedule loading (one for each simulation run)
- Labor configuration file
- Resource configuration file.

The primary sources for generating data for the simulation are Microsoft Excel and project management applications such as Primavera or Microsoft Project. The project management data for the simulation includes project task decompositions, precedence relationships between tasks, resource requirements, timing data such as due dates and duration, other scheduling data such as

project loading for the shipyard, project due dates, production area assignment options, and resource configuration data.

The preprocessor is a NIST-developed software module, written in the C programming language, that is used to translate project data into a form that can be easily read into ProModel. The data files are read into the simulator during the initialization phase of the simulation run.

2.3 Shipbuilding Control Logic

Most manufacturing simulations concentrate on modeling material flow. The shipyard simulation is different in that it concentrates on modeling the flow of work through various planning and processing stages. In the shipbuilding simulation discussed here, jobs and tasks are defined as logical entities in the simulation flow to simulate the production process. Each job is decomposed into many tasks. The following subsections discuss the detailed control logic.

As showed in Figure 2, Job orders are created and placed at a *Jobs_Arrival* location. Once the job data is initialized from data files, it is moved to the *Unstarted_jobs* location.

At the *Unstarted_jobs* location, if current date is later than or equal to the job start date, the job is moved to the *Job_area_selection* location. Otherwise, the job remains at the schedule location waiting for its start date to arrive.

At the *Job_area_selection* location, the production area where the job will be processed is determined. Production areas are assigned in order of most - late to least-late job. A lateness attribute is set on the job by subtracting the start date from the current date. The *Job_area_selection* location will attempt to select a production area for the most - late job first. The required space for the primary production area is computed. An attempt is made to allocate the required space in the primary production area. If the allocation is successful the job moves to the *Task_initialization* location for the selected production area. There is a unique set of "task " locations that is associated with each major production area. If the allocation of a production area is unsuccessful, the job remains at the J*ob_area_selection* location until sufficient free space can be found in an appropriate production area for the job. The system rechecks for free space each simulation clock cycle.

The subroutines in this module are responsible for dynamically allocating and releasing chunks of workspace within the production area. The space required is determined by job length and width data for the job that is contained in an external data file.

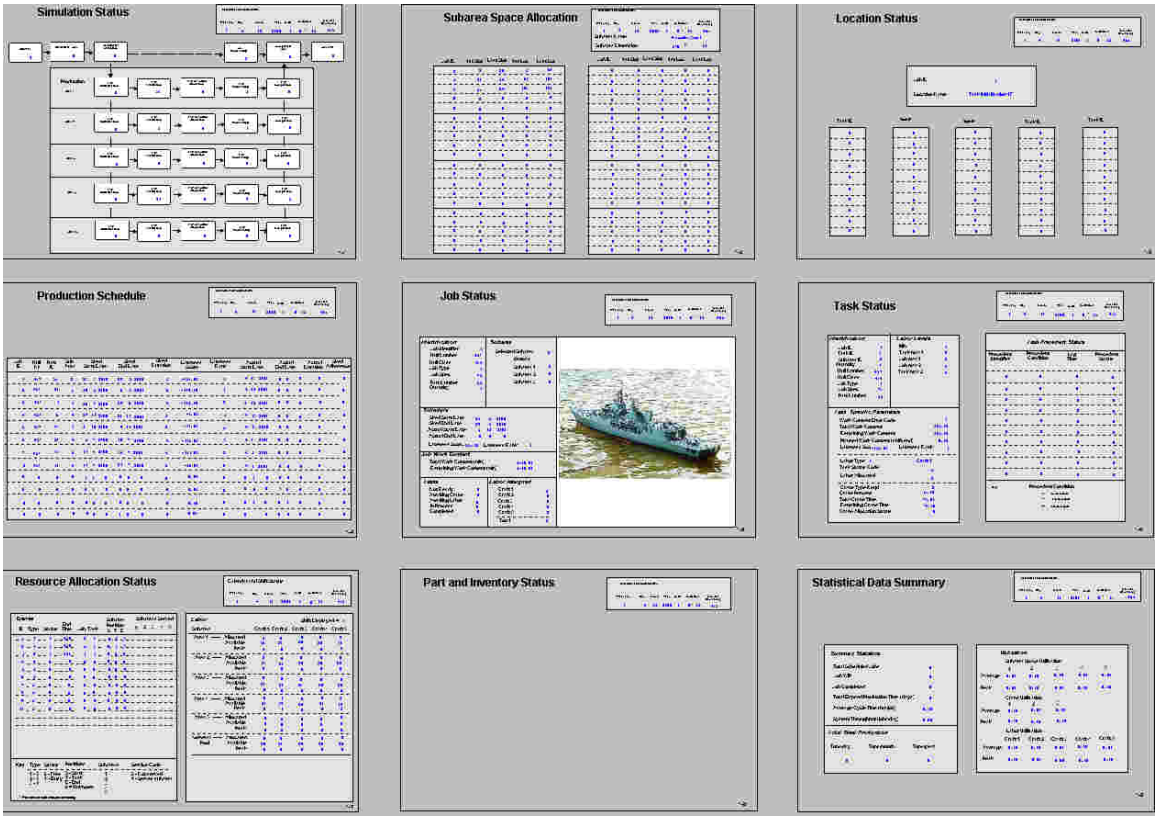

Figure 1: Shipyard Simulation User Interface

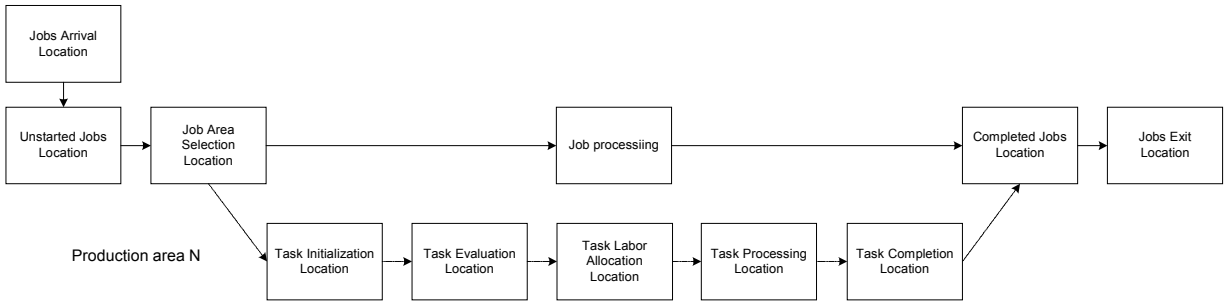

Figure 2: Entity Processing Locations within the Shipyard Simulation

At the *Task_initialization* location, another data file is used to identify the task decomposition of the job, the labor requirements, and bill of materials. The production area that has been selected and the job type determine which data set will be used. A precedent network array is filled in for all of the tasks in the job decomposition. Work duration is computed for each task in the job. Random number generators and constraint information is used to generate varying durations for each task. Totals are generated for the entire job. A unique task entity is created for each task in the job. Task attributes are set and all entities move to the *Task_evaluation* location for the selected working area.

At the *Task_evaluation* location is where a determination is made as to whether the task is ready to start. Readiness is determined by whether or not there are any precedent conditions that are not satisfied. Precedent conditions include the satisfaction of overlapping task constraints (e.g., start-to-start relationships). Ready tasks are moved to the *Task_labor_allocation* location. Tasks that are not ready remain at the *Task_evaluation* location until their precedent tasks have completed. Tasks remaining at this location are re-evaluated every clock cycle.

Each task is defined by its total work duration, work duration remaining, the labor skill category required, staffing constraints, number of workers currently assigned, and material handling (crane) requirements. These parameters are stored on the task - entity and/or task - data arrays. Work duration is maintained as the total number of minutes required to complete the task. Remaining work content refers to the work that still must be performed. Remaining work content is updated as appropriate at the start of task processing, task completion, and/or at the end of the shift. Number of workers assigned refers to the number of workers allocated to the task for the current shift. The number may change from shift to shift depending on the relative lateness of the task. Crane requirements are specified as a percentage of the total duration beginning at the start of task.

Four types of precedence relationships may be defined in the project management data file:

- Finish-to-start (FS)
- Start-to-start (SS)
- Finish-to-finish (FF)
- Start-to-finish (SF).

Given tasks A and B, the relationship FS means that task B cannot start until task A finishes. The relationship SS means that task B cannot start until task A starts. FF means that task B cannot finish until task A finishes. SF means that task B cannot finish until task A starts. Furthermore, a relationship may be indicated as one of the following types plus or minus a percentage of the task duration. For an example, an SS + 50% relationship indicates that task B can start after 50% of the duration since the start of A has passed.

At the *Task_labor_allocation* location, workers are allocated from the labor pool to each task. Priorities for labor allocation are determined by the lateness of the job. Labor allocation decisions are revisited for all jobs' tasks within a production area at the beginning of each shift. Labor is allocated to tasks at the location at the beginning of each shift. Labor allocated to a task will remain on the task until either the task completes or the shift ends, whichever comes first. When tasks complete, labor is reassigned to the pool until it is allocated to the next set of tasks that are ready to start or continue. Low priority tasks may not receive a new allocation in subsequent shifts. At the end of each shift, all tasks in the working area are routed back to the *Task_labor_allocation* location. This logic ensures that jobs that have recently become late or gotten ahead of schedule will receive an appropriate labor allocation.

Various scoring mechanisms for evaluating job lateness and performing resource allocation have been implemented. One mechanism is described here. In this mechanism, lateness is determined by comparing the percentage of work content that has completed against the percentage of the job duration that has passed. Labor pool data, i.e., availability of workers in each labor pool, is

initialized at startup from an external file. Calculations used to determine lateness are as follows:

- Scheduled job time = Job due date – Job start date
- Percent job time elapsed = (Current date – Job start date) / Scheduled job time %
- Percent job completed = 100% – (Work content remaining / Total work content)%
- Lateness score = Percent job time elapsed – percent completed.

For example, a *Lateness_score* of 100% would mean that the job is past due and no work has been performed. On the other hand, a *Lateness_score* of –100% would mean that work is completed, but its scheduled start date has not arrived. A *Lateness_score* of 0% means that the job is on schedule.

All tasks that received a labor allocation are moved to the T*ask_processing* location. Tasks that did not receive a labor allocation remain at the location until labor resources can be allocated.

At the *Task_processing* location, the task waits until either the work content is completed or the shift ends. The current time to complete the work content is determined by dividing the total task work content by the number of workers assigned. If the work completes before the end of the shift, the remaining work content for the task is set to zero and the task is moved to the *Task_completion* location. If the shift ends before the work is completed, the work accomplished during the shift is subtracted from the remaining work content and the task is moved back to the *Task_labor_allocation* location. The work accomplished during the shift equals the shift duration multiplied by the number of workers assigned.

2.4 Shipyard Input Data Files Processing

This module converts shipyard data files into appropriate formats for initialization of internal simulation data structures. The shipyard input data for the simulation may initially reside in Microsoft Excel spreadsheets, Microsoft Project files, Primavera project files, Microsoft Access databases, and Comma-Separated Value (CSV) text files. The data types used in shipyard source data files or databases are not the representation required for internal use within the ProModel simulator. The data files are preprocessed to put them in a format of the internal representation that is used to represent the data within ProModel. In this version of ProModel, all data must essentially be represented as an integer or real numbers.

Input data conversion utilities are implemented using C, Microsoft Access, and Microsoft Excel macro programming capabilities. The data file used for input into ProModel is structured to facilitate the use of its input functions, i.e.; it is converted to appropriate integer and real formats. The following major sets of data are required to configure and run the shipyard simulation models:

- Schedule file
- Shipbuilding project plans
- Labor configuration data
- Calendar and shift information.

Each of the data sets is briefly described below.

Schedule file – The schedule is used to identify the jobs to be run during a particular run of the simulation. Two schedule file formats have been implemented. The first format supports construction jobs within the shipyard. The second format supports repair and conversion jobs at shipyard piers. The schedule file is maintained in comma-separated-value (CSV) format. For example, the repair and conversion file format is a table that contains: the project id number, scheduled start day, month, year, scheduled end day, month and year, ship length, width, draft, and three possible pier assignments in priority order.

Shipbuilding project plans – Project planning data for the shipyard is maintained in a project management file that contains task identifiers, activity descriptions, total staff hours, remaining staff hours, early start date, early finish date, craft code, staff per shift, a calendar identifier, and precedence relationships between activities. A unique identifier is assigned for each project and job (subproject or group of activities/tasks). The identifier is used to reference the input data files and track the project/job through the simulation.

Labor configuration data - The shipyard configuration data file is used to identify shipyard labor, material handling, machine, and space resources that are available during a particular simulation run. The file identifies the number of each craft type and skill level that is available on each shift during a typical day. It specifies labor rates by craft type and skill level, whether new hires or contractors may augment labor pools, and simple rules for laying-off idle workers. An Excel spreadsheet may be used for data entry. ASCII CSV files are exported from Excel and read into the ProModel simulation.

Calendar and shift information - This data is currently updated using the development interface. It includes production calendar information, i.e., holidays, shift schedules, overtime constraints, etc.

2.5 HLA Interface

The High Level Architecture (HLA) was developed by the U.S. Department of Defense's Defense Modeling and Simulation Office (DMSO) to provide a consistent approach for integrating distributed, defense - department simulations (Kuhl 1999). In this project, the HLA integration mechanisms were used to manage communication and synchronization among simulation

models. Integration is achieved through the use of the NIST-developed High Level Architecture (HLA) adapter. The HLA adapter is a software module that acts as an interface between the simulations and the HLA Run-Time Infrastructure (RTI). The NIST HLA adapter provides a simplified interface for using the HLA RTI communication mechanisms.

An HLA-based distributed simulation is called a federation. Each simulation application that is integrated by the HLA RTI is called a federate. One common data definition is created for domain data that is shared across the entire federation. It is called the federation object model (FOM). Each federate has a simulation object model that defines the elements of the FOM that it implements (McLean and Riddick 2000).

A DMS Adapter Module is incorporated into each DMS federate. The DMS Adapter handles the transmission, receipt, and internal updates to all FOM objects used by a federate. Figure 3 illustrates the relationship between the various elements of the distributed manufacturing simulation execution environment. For the shipyard simulation, multiple visualization federates were implemented to run with a single simulation engine and manufacturing model. The DMS Adapter and the HLA RTI provided remote updates of shipyard simulation data to remote user interface screens.

3 CONCLUSIONS

This document has provided a brief overview of the simulation of shipbuilding operations that is being developed as a part of the NSRP Project by staff of the NIST Manufacturing Simulation and Visualization Program. By using simulation in shipbuilding operation, traditional problems in a shipyard can be solved. The simulation model enables the analysis of

- Schedule impact due to additional projects
- Differing production scenarios and identification of Labor resource problems
- Identification resource constraints and conflicts between competing jobs (including cranes and forklifts)
- Demonstration of the expected results of inserting new technology or equipment into shipyard, particularly with respect to operating costs and schedule impact.

The animated and graphical display of work location, resources, production flows, and simulation results will help shipyards to make better production management and resource allocation decisions. Formatted report and log files archive the simulation runs to preserve historical data for later use.

The integration with the other simulation models using DOD High Level Architecture and Run Time Infrastructure as an integrating infrastructure currently enables remote interfaces and will ultimately enable the implementation of larger and larger distributed models of the entire shipyard manufacturing system.

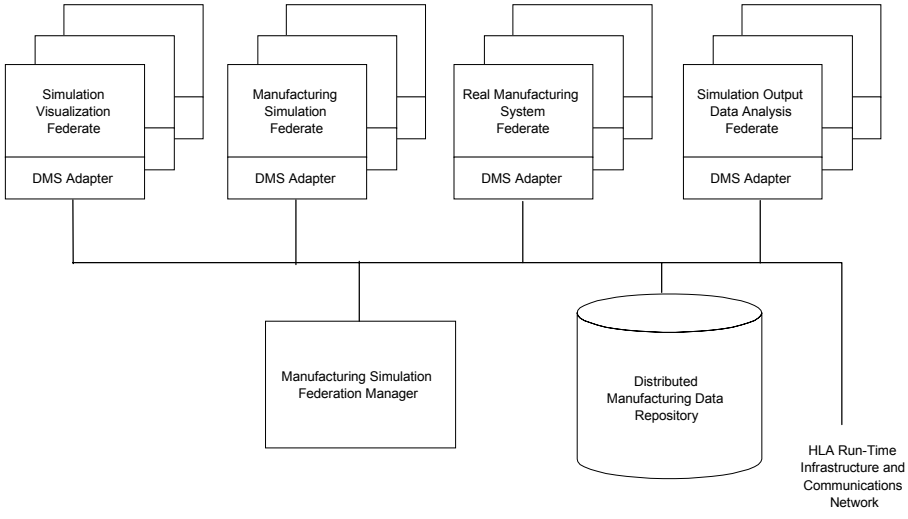

Figure 3: Distributed Manufacturing Simulation Environment Elements Integrated by the HLA Run Time Infrastructure

DISCLAIMER

No approval or endorsement of any commercial product by the National Institute of Standards and Technology is intended or implied. The work described was funded by the United States Government and is not subject to copyright.

ACKNOWLEDGMENTS

Work described in this paper was sponsored by the National Shipbuilding Research Program (NSRP) ASE and the NIST Systems Integration for Manufacturing Applications (SIMA) Program. The authors would like to express thanks to Mike Iuliano, Young Jun Son, Khadija Moultalis, Swee Leong, Tina Lee and Saumitra Chopade for their efforts in the project.

REFERENCES

Kuhl, F., R. Weatherly, and J. Dahmann. 1999. Creating Computer Simulations: An Introduction to the High Level Architecture. Prentice Hall: Upper Saddle River, NJ

McLean and Riddick. 2000. The IMS MISSION Architecture for Distributed Manufacturing Simulation. *Proceedings of the 2000 Winter Simulation Conference*, Edited by Joines J. A Barton R. R. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers. Or via ‹http:// www.informs-cs.org›

ProModel Manufacturing Simulation Software Reference Guide (PRG), 1996, Orem, UT

ProModel Manufacturing Simulation Software User's Guide (PUG), 1996, Orem, UT

Storch, Hammon, Bunch and Richard C. 1995. Ship Production, 2nd Edition. Cornell Maritime Press, Inc.

The National Shipbuilding Research Program (NSRP) WebPages <http://www.nsrp.org>

AUTHOR BIOGRAPHIES

CHUCK MCLEAN is Leader of the Manufacturing Simulation and Visualization Group in the U.S. National Institute of Standards and Technology (NIST) Manufacturing Systems Integration Division. He has managed research programs in manufacturing simulation, engineering tool integration, product data standards, and manufacturing automation at NIST since 1982. He has authored more than 50 papers on topics in these areas. He holds a Master's Degree in Information Engineering from University of Illinois at Chicago and Bachelor's Degree from Cornell University. His e-mail address is <mclean@cme.nist.gov>

GUODONG SHAO is a research engineer from Intelligent Automation Inc. and a contractor in the Manufacturing Simulation and Visualization Group in the U.S. National Institute of Standards and Technology (NIST) Manufacturing Systems Integration Division. He has participated in research relating to FMS, CIMS, and manufacturing simulation integration for years. He holds a Master's Degree from University of Maryland at College Park. He is a Ph.D. student in Graphics Laboratory at George Mason University. His e-mail address is <gshao@cme.nist.gov>