Proceedings of the 2009 Winter Simulation Conference
M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, eds.

YARDSIM: A RAIL YARD SIMULATION FRAMEWORK AND ITS IMPLEMENTATION IN A MAJOR RAILROAD IN THE U.S.

Edward Lin

Norfolk Southern Corporation
1200 Peachtree St. NE
MS 12-117
ATLANTA, GA,30309,USA

Clark Cheng

Norfolk Southern Corporation
1200 Peachtree St. NE
MS 12-117
ATLANTA, GA, 30309, USA

ABSTRACT

Rail is the most environmentally friendly and fuel efficient mode of freight transportation. Classification yards are railroads' single-most complex operation. Railroad operations are directly influenced by their performance. In this paper, we present a rail yard simulation framework, or YardSim, and describe how it has been used to model a major hump yard on a Class I railroad in the US. The framework, that models complicated rail yard operations from inbound arrivals, inspection, switching, assembling, brake test, to outbound departures, can be used to pinpoint bottlenecks, identify improvement opportunities in operating practice and yard infrastructure, and assess the impact of changes in traffic volume and service plan. The YardSim provides the flexibility and adaptability to conduct and manage "what-if" analysis on train plan, yard infrastructure, operating policy, operational strategy, and yard resources such as crews and locomotives. Therefore, the development cycle for a new yard can be reduced significantly.

1 INTRODUCTION

Railroads are one of the most environmentally friendly and fuel efficient modes of freight transportation, transporting a ton of freight 423 miles on a single gallon of fuel (Railway Age, February 2009). One train can take up to 300 truckloads of freight off our congested highways. Trains can improve air quality and dramatically reduce greenhouse gas emissions. That helps reduce pollution, fuel use and demand for foreign oil. Rail yards represent critical points in the rail network. Vantuono (2009) states that "classification yards are the railroads' single-most complex operation, where local problems tend to ripple across the network work." Every day, hundreds of cars arrive at a classification yard, which are separated, sorted into new blocks according to their destination, and assembled to form a new outbound train. Petersen (1977) identified several types of yards that exist in most railway systems. Hump yards are the largest and most complex yards. However, they are also the most efficient yards to classify cars. A typical hump yard may be several miles long and up to one mile wide. Within a yard, there may be more than 100 miles of tracks and hundreds of track switches.

Yard operations require variable resources, such as switch locomotives, switch crews, car inspectors, and mechanical personnel to repair broken railcars. Based on various studies for the rail industry, an average rail car spends about one third of its system time on the main line and two-thirds within terminals (Reebie Associates 1972, McKinsey & Company 1992, Logan 2006, and Mercer Management Consulting 2006). For every 15% reduction in system-wide average terminal dwell time, there is an approximate increase in carload velocity of 2 mph. The efficiency of yards affects the effectiveness of the rail network. It is therefore important to evaluate yard performance with respect to effectiveness and efficiency.

The challenging of operating a yard lies in coordinating efficient movements of cars in the yard, given limited resources (such as track capacities, switch crews, switch locomotives, car inspectors), train schedules, and traffic priorities. Different decision making methods, resource allocation strategies and train plans will affect the efficiency and timeliness of yard operations. Due to the highly complex yard operations, the application of exact mathematical optimization methods is quite limited. Computer simulation, combined with mathematical programming models, offers a flexible and credible technique to identify opportunities for yard performance improvement.

A rail yard simulation model prescribes a set of operating policies and operational strategies to simulate yard activities. Each rail yard has different infrastructure and unique operating characteristics. An operating policy for one rail yard may not

be applicable to another rail yard. Consequently, the best operational strategy of one rail yard may not be suitable to other yards. Thus, yard simulation software requires the capability of easily incorporating different yard infrastructure, operating policies, and operational strategies.

Several commercial simulation software packages indicate their capability of modeling rail yard operations. Flexsim CT of Flexsim is a simulation library that can be used to build container terminal operations simulation model. Flexsim is developed in C++ language. AnyLogic of Xj technologies is a multi-method simulation software and develops a simple classification yard that uses its rail yard library to demonstrate its rail yard modeling capability. Anylogic simulation model can be invoked through web interface. Villon of Simcon is a simulation tool that can be used to build detailed simulation model of logistic junction and its technological processes. Villon Model is a tool for the rationalization and optimization of operations in a logistic node. As the software is very complex, the user must be trained to use the system. The challenge of a rail yard simulation model is the needs for modeling detailed activities. Based on our investigation, it requires tremendously amount of internal resources and efforts and, in some cases, external consulting services, if not impossible, to customize commercial software to handle different rail yard infrastructure and yard operational strategies. At the end, we decided to develop a rail yard simulation framework in house from ground-up.

The objective of this paper is to present a framework that can be easily adapted to model different rail yards. As a result, the development cycle time can be reduced significantly. In the following sections, we present our approach to develop a rail yard simulation to manage yard resources and rail car movements in hump yards. First, we give an overview of typical operations in a rail hump yard and common methods for managing yard operations. Then, we discuss YardSim, a rail yard simulation framework. Next, we describe our implementation of the rail yard simulation framework in a hump yard on a Class I railroad in the US. Finally, we conclude our development and implementation and next steps.

2 BACKGROUND

In railroad freight transportation, a shipment to be moved from its origin to destination is carried by a rail car that meets its characteristics. When demand is high, a set of shipments with the same origins and destinations are grouped together in blocks and moved directly from origin to destination. However, when demand is not sufficient, shipments are consolidated and carried through a sequence of trains. Typically, cars with different final destinations but sharing a portion of their trips are assembled into blocks. Cars in the same block travel together in railroad network in order to increase efficiency and reduce intermediate handlings. At a transfer location, cars are re-sorted for their next destination in the trip. The locations where cars are sorted and regrouped into blocks are called classification yards. There are two types of classification yards: Hump yards and Flat switching yard. Flat yards are more common in U.S. railroads; however, hump yards are much more efficient. Figure 1 shows a hump yard that contains a receiving yard where trains are received, a classification yard (also known as a bowl), a forwarding yard where outbound trains are assembled and depart, and pull back tracks that are used to move cars from classification yard to forwarding yard

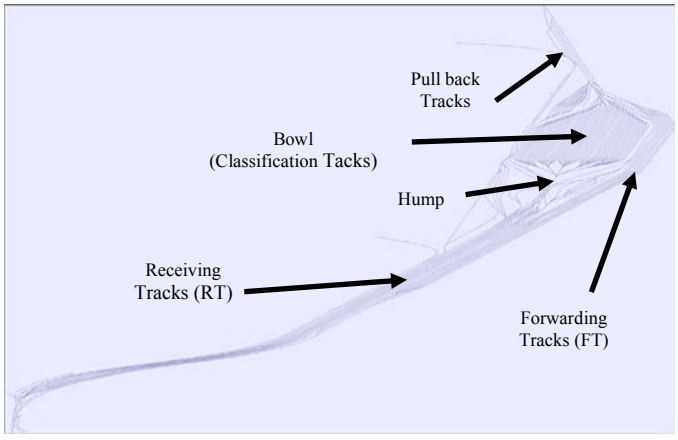

Figure 1: A hump yard

When an inbound train arrives at a hump yard, the terminal trainmaster instructs the train crew to park cars in the receiving yard, the cars may go through inbound car inspection process at the receiving yard, and then the air on each car is bled out to release the air brake. Next step is humping, a process in which a locomotive (also known as an engine) pushes cars from a track in the receiving yard over the hump. When cars arrive at the hump, the highest point of the hill, pins are pulled

and cars roll under the force of gravity to assigned classification tracks in the bowl. Retarders are used to control the car speed, so they will have enough velocity to reach their assigned track, but not damage the already parked cars. Depending on shipment characteristics and destinations, several cars may be roll together as a group. Ideally, each classification track is designated to an outbound block. But due to the limited number of classification tracks, several outbound blocks can be assigned to one classification track. Those cars may need to be re-classified (or re-hump) later on when tracks become available. Cars are waiting on the classification tracks for departure trains. When it is time to assemble an outbound train, outbound blocks are pulled from classification tracks and assembled into the outbound train. The list of potential blocks that each outbound train can carry is specified in the operating plan. The sequence of blocks on an outbound train is called block standing order. The blocking standing order determines the sequence of cars that need to be pulled from classification tracks. A pull back engine first pulls the cars for the first block onto a pull back track, and then shovels them to a departure track in the forwarding yard. This process is repeated until all cars for the outbound train are assembled. After a train is assembled, the cars must be inspected for mechanical failure and defects and a brake test must be performed before the train can leave the yard. If a car is identified as a bad order, it must be set out from the train and sent to a bad-order track for repair.

Figure 2 shows a typical process flow in a hump yard. Rectangular boxes represent processes and triangular shapes represent receiving/classification/forwarding yards or buffers. Dirnberger and Barkan (2006) noted from literatures that majority of time cars spent in terminal (about 77.3%) is non-value-added idle time, waiting for next process.

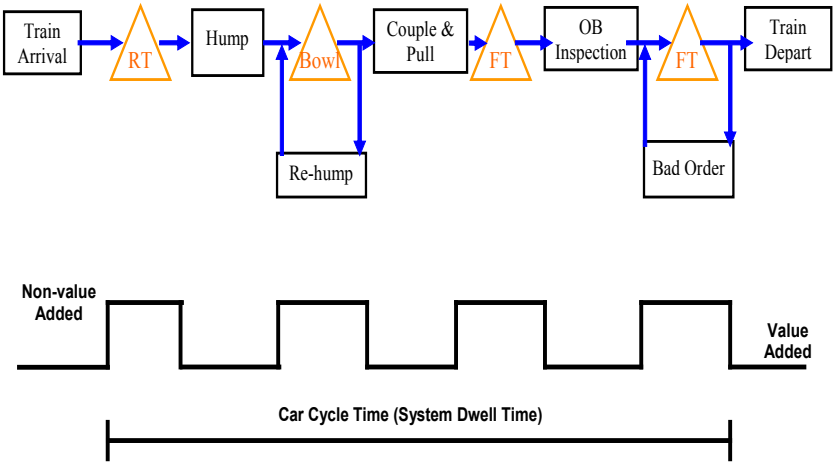

Figure 2: A process flow in a hump yard

There are a number of decisions to be made in each process to ensure that cars are transported efficiently and safely to meet yard performance measures and resources are fully utilized. In the following, we summarize important decisions that are made in each process and methods found in literature.

1. Train Arrival Process
   - Inbound Train Route
     Generally, there are several alternative routes to bring a train into a yard. An Inbound trains may be blocked by humping or pull back activities. The objective of inbound train route selection is to minimize the impact on yard activities, especially humping activity while avoid or minimize the time waiting on the mainline outside the yard. Inbound train route is determined by trainmaster when it approaches the yard.
   - Receiving Track Assignment
     Several factors dictate the assigning of a receiving track(s) to a train. The factors include available track length, train length, and inbound train approaching direction. If train length is shorter, the train can double with other trains on one track. If train length is longer than the assigned track length, the train will be split and occupies more than one track.
2. Humping Process
   - Humping Sequence
     Blocks of cars at receiving tracks can not be moved unless classification tracks are available. Humping sequence determines the order of trains to be humped. Yagar, Saccomanno, and Shi (1983) suggested a dynamic

programming approach as well as a screening technique to optimize sorting and assembly operations. Kraft (2000) studies the hump sequence problem with a real-time planning perspective.

- Classification Track Assignment
  The block-to-track assignment problem has been extensive studied in literature because it is the most widely recognized perspective to improve the blocking capacity of a yard. There are two types of block-to-track assignment strategies: static and dynamic. Block-to-track assignment does not change in the static assignment strategy. In dynamic block-to-track assignment strategy, block-to-track assignment is allowed to vary over time (as known as swing) as traffic volume and mix change. Daganzo (1983), Daganzo (1986) described several multistage sorting strategies: sort-by-train, sort-by-block, geometric sorting, and triangular sorting. Each classification track is assigned several blocks and cars are resorted during train build up. The relative performance of different multistage sorting strategies was investigated by Daganzo, Dowling, and Hall (1983). Dynamic block-to-track assignments are studied in Kraft (2002a) and Kraft (2002b). Dirnberger and Barkan (2006) proposed a sort metric to measure the quality of sorting strategies.

3. Couple and Pullback Process
   - Outbound Train Route
     Most of the freight railroads have shifted from a tonnage-based operating strategy to a schedule-based operating strategy. On-time train departure becomes an important performance measure. Due to congestion in a yard or on a mainline, a train may depart using alternative route. Outbound train route is determined when the outbound train is ready to assemble. The route may change the order in which cars have to be pulled from classification tracks.
   - Forwarding Track Assignment
     Forwarding track is the place to build outbound trains and perform mechanical inspection before a train departs. One of the most deciding factors is yard infrastructure and its connectivity to mainline. Other factors include track length, air pump station, and mechanical crew station.
   - Pulls and pull sequence
     Locomotive is used to pull cars from classification tracks to forwarding track via pull back tracks. Locomotive horsepower and pull back track length limit the number of cars can be pulled out of classification tracks at one time. There are two methods of making up trains: single pull per engine and multiple pulls per engine. In multiple pulls per engine cars are pulled from several classification tracks before transport cars to a pull back track.

3 RAIL YARD SIMULATION FRAMEWORK

Due to the complexity of yard operations and management, a rail yard operations planning and management tool to support yard capacity and improvement studies is needed. A yard capacity study is to assess how many cars, blocks and trains can be handled with the existing infrastructure and resources. Rail yard improvement studies may include infrastructure improvement (e.g., adding a new track or crossover), resource requirements (e.g., number of yard engine and crews), and capability to handle increased traffic and trains. Thus, a rail yard simulation framework should be highly re-usable for different yard layouts, customizable to handle different operating policies and scheduling strategy, adaptable to organization structure and responsibility, and scalable to handle changes in traffic volume and mix, train plans and resource levels.

Rail Yard Simulation framework includes a rail yard simulator, a set of support applications, and libraries to help develop and assemble together different components of a simulation model. The support applications include:

- Input data bridge: import and edit train consists and train schedules;
- Yard layout editor: import and edit yard layout and infrastructure data,
- Process flow and business rule engine: import and manage operating procedures and scheduling strategy.
- Scenario manager: a decision support system is also required to manage different scenarios from the "what-if" analysis.
- Report generator: calculate key statistics and metrics and generate management reports

Figure 3 shows typical inputs and outputs of a rail yard simulation model.

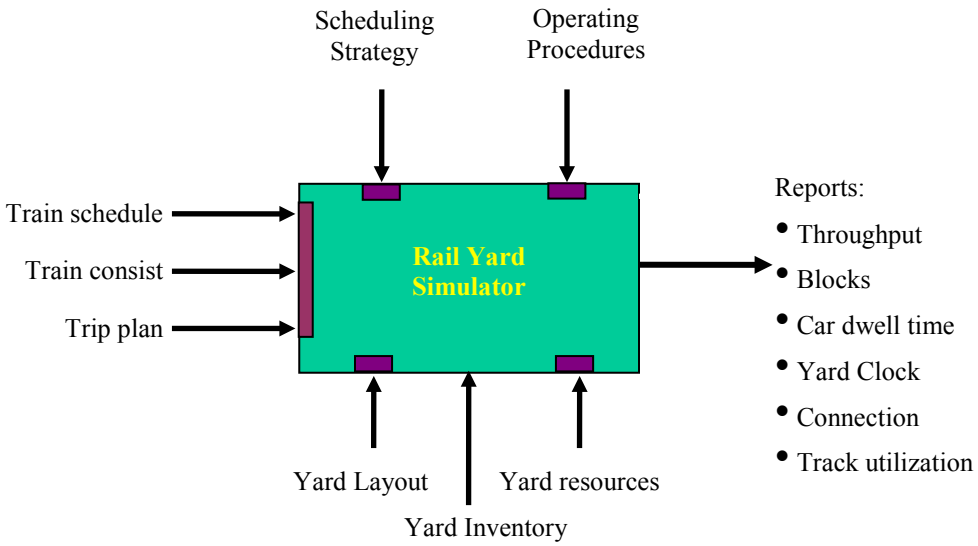

Figure 3: Rail yard simulator

Rail Yard Simulator is the core component in the framework. A rail yard simulation model represents critical activities that are necessary for yard improvement analysis. Typical activities include car movement, switching track, hookup (coupling), cutoff (de-coupling), brake test, inspection, humping, and bad order handling. Depending on track condition, switch turnout, regulation, and policy, car movement speed varies within a yard. Railroad switches are mechanisms used to guide car movement from one track to another track at a railway junction. A detail description of railroad switch can be found in <http://en.wikipedia.org/wiki/Railroad_switch>. Figure 4: Car movement shows a typical car movement pattern. To move cars from Track A to Track C, the car movement path is Track A – Track B – Track C. If locomotive is in front of cars along the car moving direction, it is a pull move. On the other hand, it is a push move if locomotive is behind the cars along the car moving direction. The distance that the cars need to go on the track B depends on the total car length and track clearance distance. Before cars can start moving from Track B to Track C, conductor needs to throw switch to align Track B to Track C and lock the switch. For example, it is a pull move from Track A to Track B and a push move from Track B to Track C. Thus, detailed simulation of car movements is required. Since there are many concurrent car movements in a yard, 3D animation will be a very useful visualization tool for model validation and demonstration.

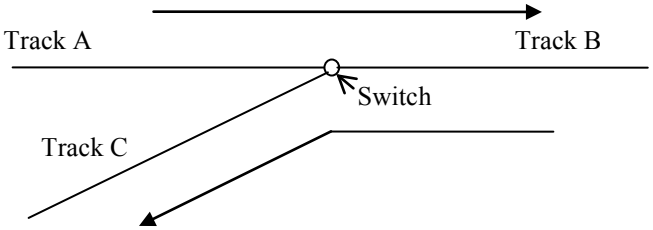

Figure 4: Car movement example

Figure 5 shows another important car movement behavior. It shows two car movement paths: Track D – Track E – Track F and Track G – Track E – Track H. Track E is common in both paths. There is no traffic light in a rail yard. To move cross the common track requires permission from responsible yard master to throw switches. This behavior is noted as an operating policy and the Rail Yard Simulator is flexible to plug-in different operating policies.

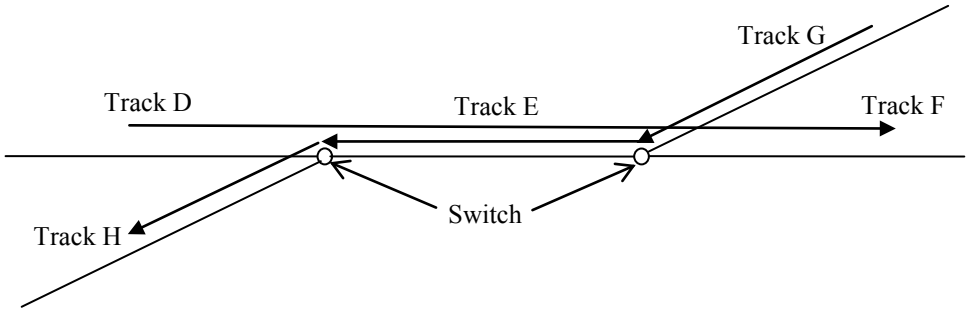

Figure 5: Car movement at crossover

A rail yard library, a repository of reusable components, contains yard layouts, yard resource profiles, sets of operating policy and scheduling strategy, and traffic mix. The library is defined as a 6–tuple (L, R, O, T, W1, W2), where

- L is a finite set of yard layouts
- R is a finite set of resource profiles
- O is a finite set of operating policy and scheduling strategy
- T is a finite set of traffic/train plans
- W1: (O * R) → N defines the relationship between yard resource profiles and sets of operating policy and scheduling strategy. N ∈ {0, 1}.
- W2: (O * T) → N defines the relationship between traffic mix/train plan and sets of operating policy and scheduling strategy. N ∈ {0, 1}.

A rail yard simulation model is defined as <l, r, o, t>, in which l ∈ L, r ∈ R, o ∈ O, t ∈ T, W1: (o * r) → 1, and W2: (o * t) → 1. To build a simulation model is to assemble components from rail yard library. The Rail Yard Simulator is capable of handling different rail yard simulation model. In order to achieve this goal, the Rail Yard Simulator defines a set of interfaces.

Support programs include applications to create yard layouts, build operating policy and scheduling strategy, and resource profiles.

4 YARDSIM IMPLEMENTATION

YardSim is an implementation of the rail yard simulation framework. YardSim includes Rail Yard Simulator (a rail yard simulator), Rail Yard Editor (a rail yard layout editor), Rail Yard Modeler (a rail yard operating policy and scheduling strategy designer), and Yard Scenario Manager (a rail yard scenario management decision support system). In the following, we describe key functionality of each component.

4.1 Rail Yard Simulator (YardSimlator)

Rail Yard Simulator is the central component in the rail yard simulation framework. We developed a Rail Yard Simulator in Java.

Figure 6 shows a 3-level hierarchical control structure in the YardSimulator. The lowest level is a discrete-event simulation. The middle level receives processes from scheduling level and send proper instructions to the simulation. A process is a workflow that depicts a sequence of tasks. This level also receives feedback from simulation. Feedback includes acknowledge signal, task complete signal, etc. The top level schedules processes and manages resources and sends process details to the task execution control. It receives task completion feedback from the task execution control. The top level receives incoming trains and outgoing trains and schedule processes to inbound or outbound trains.

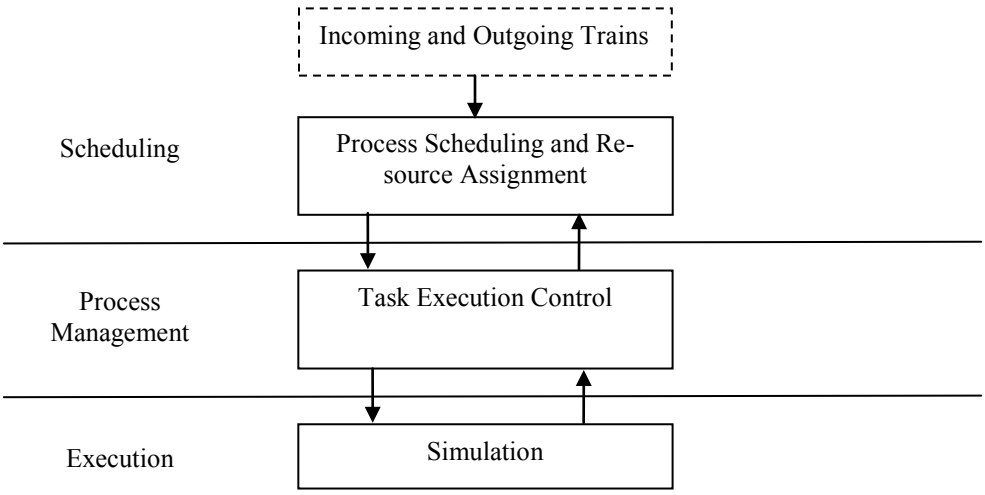

Figure 6: A yard hierarchical control architecture

The YardSimulator includes the following components: simulation engine, 3D animation engine, process engine, statistic engine, and library of business rules, heuristics and algorithms. Simulation engine is a discrete event simulation engine. An open source discrete event simulation engine is used at execution level to manage future event list. We developed a Java 3D animation engine. The animation engine receives yard layout data and car position data and display them in Java 3D. An open source process flow and rule engine is used to execute tasks. A task execution control Java program is developed to manage communications between scheduling and execution levels. The task execution control Java program also coordinates with process flow and rule engine to execute task.

In order to move cars from one location to another location, a car routing path needs to be identified. We developed an enhanced shortest path algorithm that considers car length and different types of movement, push or pull. Movement instructions (or recipes) are sent to simulation. Simulation interprets recipes and generates proper simulation events. Figure 7 is a 3D animation screenshot.

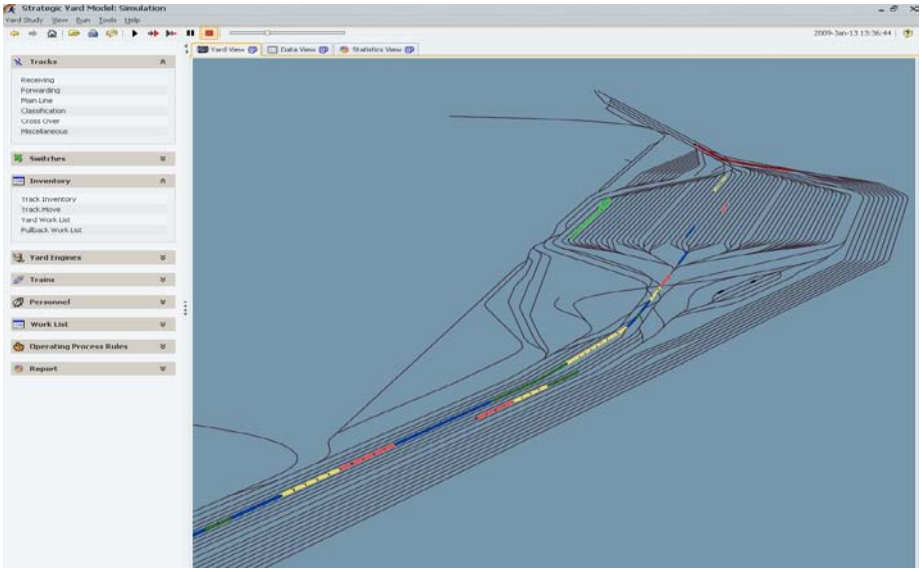

Figure 7: 3D animation screen

4.2 Rail Yard Editor – Rail Yard Layout Editor

Rail Yard Editor is a software tool used to edit yard layout. CAD drawings are commonly used to represent yard layout. The Rail Yard Editor provides functionality to transform electronic yard geometry data in AutoCAD format into the format that

could be used in the Rail Yard Simulator. It also provides switch labeling, track labeling, track functional areas assignment, track distance measuring features.

The Yard Layout data to the Rail Yard Simulator is represented as G= (V, A, S, K, P, W3, W4), where V is a set of nodes (or points), A is a set of lines, S is a set of switches (S ⊂ V), K is a set of tracks, P is a finite set of ordered lines, W3: (K * K) → N defines feasible immediate movement between a pair of tracks, and W4: (K * P) → N defines the Polyline of a track. Figure 8: Yard layout data generation processshows the steps to generate yard layout data.

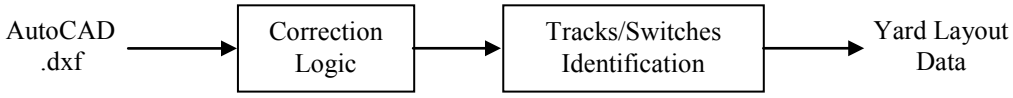

Figure 8: Yard layout data generation process

The geometry data of an AutoCAD drawing in .dxf format is processed by a java program, which uses correction logic to reduce precision to get a better connectivity, to explore trimesh points, and to linearize curves. A depth-first search algorithm was developed to identify tracks, switches, and polylines associated with each track.

Yard Layout Editor reads the yard layout data and allows users to assign track name, switch name, and track functional area. A utility is also developed to store the yard layout data in a DB2 database. The yard layout data can be used to build a simulation model.

4.3 Rail Yard Modeler – Rail Yard Process Flow and Business Rules Designer

Rail Yard Modeler is used to develop operating policy and scheduling strategy. The Rail Yard Modeler supports real-world processes based on workflow and organizational model. Currently, we use an open source process flow and rule engine to implement operating policy and scheduling strategy. A role-based approach is used to model yard operation policy and scheduling strategy. In this approach, each role is modeled as an object with prescribed responsibilities or services. Typical roles in a hump yard include terminal train master, yard master at the hump tower, yard master at pull back tower, mechanical inspection crew, air bleeder, train crew, hump engine, and pull back engine. Each role (or object) uses push-pull strategy to process requests. A simple air bleeding process in MS Visio is shown below to illustrate the push-pull strategy.

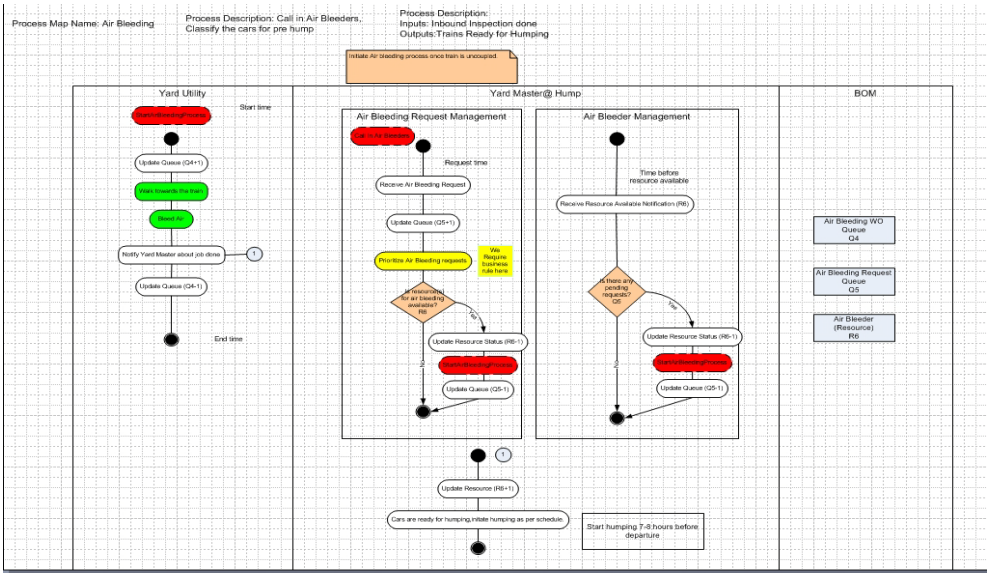

Figure 9: A simple air bleeding scheduling strategy

The process flow is described as follows:

- Yard Master at the Hump tower receives an air bleeding request

- The yard master checks if an air bleeder is available
- If so, pick an air bleeder with maximum rest time, then assign the task to the air bleeder and give instructions to the air bleeder, otherwise make a note of the request
- The air bleeder notifies the yard master and waits for further instructions when the task is done.
- The yard master receives resource available notification, checks if there is an air bleeding request in the queue
- Use FIFO rule to determine next task and notify the air bleeder to perform the task
- Otherwise, instruct the air bleeder to walk to rest location

4.4 Rail Yard Scenario Manger – Rail Yard what-if analysis Manager

The Rail Yard Scenario Manager allows users to specify simulation inputs and configure simulation parameters. It maintains various "what-if" scenarios that users can review and compare simulation results. The "What-if" capability includes the following inputs and parameters: 1) yard layout, 2) yard resource availability, 3) yard operating policy and scheduling strategy, and 4) traffic volume and train plan. It allows users to run multiple simulation models concurrently. The Rail Yard Scenario Manager provides web interfaces. Users can access it via a web browser. It reduces computing power requirements on client machines since simulation models are running on application servers.

Sometimes, a yard layout may be too large to fit in one screen. The Rail Yard Scenario Manager allows multiple clients to connect to a single simulation instance. Each of them can view 3D animation from different angle. We have successfully implemented the architecture which uses Adobe Flex and Tomcat web server. Figure 10 shows the architecture of the Rail Yard Scenario Manager.

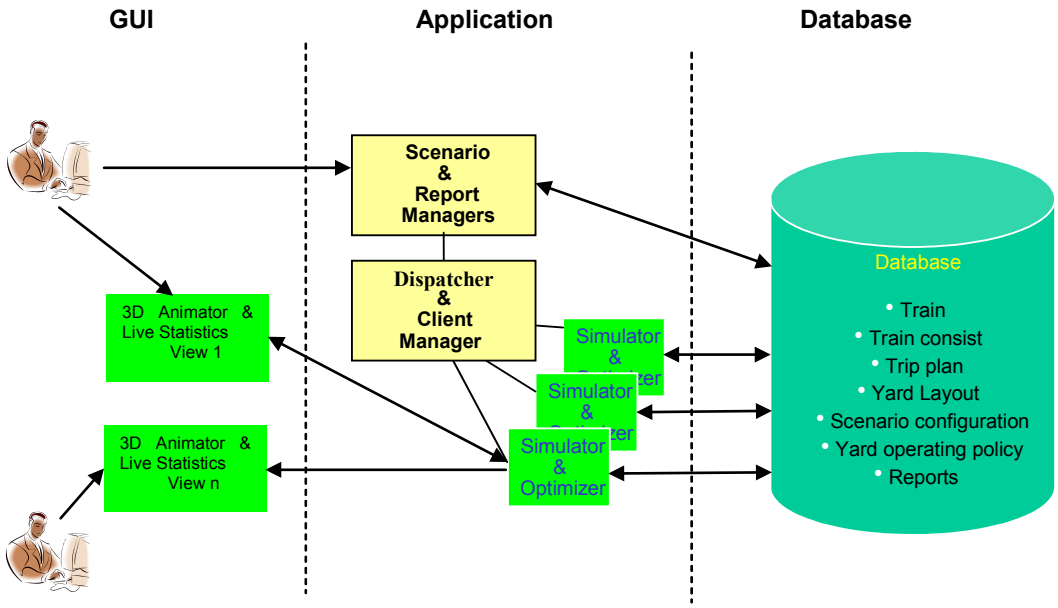

Figure 10: Rail yard scenario manager architecture

During a simulation run, the YardSimulator collects and generates many reports, such as detailed car movements, yard work orders, pull back work orders, re-humps, and inventory. These reports are imported to database and are used to verify the simulation model. A set of quality metrics are identified based on the historical data. The values or the ranges are used as the benchmark for the simulation model solution quality metrics.

5 CONCLUSION

This paper presents a rail yard simulation framework that can used to model various rail yard layouts, operating policy and scheduling strategies, and resource allocation levels. A prototype of the framework has been implemented at a major hump yard on a Class I railroad in the US. A full scale version of the implementation is being developed at the time of submission

and is expected to be completed this fall. We believe, with the framework, new algorithms and business rules can be easily plugged in and integrated with the Rail Yard Simulator. The development cycle for a new yard can be significantly reduced.

REFERENCES

Daganzo, C. F. 1986. Static Blocking at Railyards: Sorting Implications and Track Requirements, *Transportation Science*, Vol. 20, No. 3.

Daganzo, C. F., R. G. Dowling, and R. H. Hall. 1983. Railroad Classification Yard Throughput: The Case of Multistage Triangular Sorting, *Transportation Research*, Vol. 17A, No.2, 95-106.

Dirngerger, J.R. and C. P. L. Barkan. 2006. Implementing Bottleneck Management Techniques and Establishing Quality of Sort Relationships to Improve Terminal Processing Capacity. In *Proceedings of the 7th World Congress on Railway Research*, Montreal.

Flexsim. 2009. Flexsim CT (Container Terminal Library), <http://www.flexsim.co/products/ct> [accessed March 30, 2009]

Kraft, E.R. 2002. Priority-based classification for improving connection reliability in railroad yards – Part 1: Integration with car scheduling, *Journal of the Transportation Research Forum* 56:93-105.

Kraft, E.R. 2002. Priority-based classification for improving connection reliability in railroad yards – Part 2: Dynamic block to track assignment, *Journal of the Transportation Research Forum* 56:107-119

Logan, P. 2006. People, Process, and Technology – Unlocking Latent Terminal Capacity. *Transportation Research Board 85th Annual Meeting presentation*, January 24

McKinsey & Company. 1992. *Improving CBU Performance through Integrated Fleet Management: Phase I – Progress Review*. Unpublished Study for the Atchison, Topeka, and Santa Fe Railway.

Mercer Management Consulting. 2006. Asset Velocity, Terminal Performance and Netowrk Fluidity, In *MultiRail User Conference*, May 24.

Petersen, E. R. 1977. Railyard Modeling: Part I. Prediction of Put-Through Time, *Transportation Science*, Vol. 11, No. 1, February.

Reebie Associates. 1972. Toward an Effective Demurrage System, prepared for U.S. Department of Transportation, Federal Railroad Administration.

Simcon. 2009. Villon, <http://www.simcon.sk/frames.html> [accessed March 30, 2009]

Vantuono, William C. 2009. Fine-tuning the Class Yard, *RailwayAge*, March.

Xj technologies. 2009. AnyLogic – Multi-method Simulation Software, <http://www.xjtek.com/anylogic> [accessed March 30, 2009]

Yagar, S., F. F. Saccomanno, and Q. Shi. 1983. An Efficient Sequencing Model for Humping In a Rail Yard, *Transpn Research,* Vol. 17A, No. 4, 251-262.

AUTHOR BIOGRAPHIES

EDWARD LIN manages operations research projects at Norfolk Southern Corporation. He received his Ph.D. in industrial and systems engineering from the Georgia Institute of Technology. He has 15 years of research and development experience in academic, software, and transportation. His research interests include operations research and simulation applications in transportation and manufacturing industries. His email address is <edward.lin@nscorp.com>.

CLARK CHENG is the senior manager of the Operations Research group at Norfolk Southern Corporation, a Class I railroad in the U.S. Clark has been working in the railroad industry for 15 years and has participated in developing computer models for locomotive dispatching, car scheduling, network optimization, fleet planning, demand forecasting, yard simulation, crew assignments, service design, and car distribution. He received his Ph.D. degree in industrial engineering from Clemson University. His email address is <clark.cheng@nscorp.com>.