

ELIST8: SIMULATING MILITARY DEPLOYMENTS IN JAVA

Charles N. Van Groningen, Dariusz Blachowicz, Mary Duffy Braun,
Kathy Lee Simunich, and Mary Ann Widing
Argonne National Laboratory
9700 South Cass Avenue, Bldg. 900
Argonne, IL 60439, USA

ABSTRACT

Planning for the transportation of large amounts of equipment, troops, and supplies presents a complex problem. Many options, including modes of transportation, vehicles, facilities, routes, and timing, must be considered. The amount of data involved in generating and analyzing a course of action (e.g., detailed information about military units, logistical infrastructures, and vehicles) is enormous. Software tools are critical in defining and analyzing these plans.

Argonne National Laboratory has developed ELIST (Enhanced Logistics Intra-theater Support Tool), a simulation-based decision support system, to assist military planners in determining the logistical feasibility of an intra-theater course of action. The current version of ELIST (v.8) contains a discrete event simulation developed using the Java programming language. Argonne selected Java because of its object-oriented framework, which has greatly facilitated entity and process development within the simulation, and because it fulfills a primary requirement for multi-platform execution. This paper describes the model, including setup and analysis, a high-level architectural design, and an evaluation of Java.

KEY WORDS

Discrete simulation, military, deployment, transportation, Java

1. INTRODUCTION

The military logistics community has successfully used ELIST (v.7) to assist in planning analyses and training exercises for a number of years [1]. Ongoing use of ELIST7 has led to requests for more detail, more capabilities, and increased flexibility and speed. ELIST7 runs at a fairly aggregate level in which individual transports are not explicitly modeled; rather, transportation capabilities are represented in short-ton-miles/day. ELIST7 is primarily a C language tool and uses formatted files for input.

ELIST (v.8) is a completely new system for modeling the transportation of military cargo at the individual vehicle level and requires a much more detailed simulation than was required in ELIST7. The new version is intended to make it easier to integrate ELIST8 with other models and tools in the fort-to-foxhole logistical arena. ELIST8 uses the Oracle database management system (DBMS) for data persistence (rather than the flat files used in ELIST7) to facilitate data sharing with other models. Because the desired changes (especially to the simulation) were so significant, modifying the existing ELIST7 would not have been efficacious. Argonne National Laboratory selected Java as the development language because (1) it has object-oriented framework, which greatly facilitates entity and process development within the simulation, and (2) it fulfills a primary requirement for multi-platform execution. One concern in using Java, however, was scalability, which is discussed later.

2. MODEL

The model underlying the ELIST8 simulation is based on many discussions with military logisticians. A requirements document, which combines identified requirements, experiences, and observations, provides the basis for all technical design and implementation decisions [2]. ELIST8 models the intra-theater movements of personnel, equipment, and supplies by limited transportation assets over a constrained transportation infrastructure. The purpose of the system is to aid planners in evaluating the potential success of a theater leg of a transportation plan, analyzing vehicle and facility requirements for a given plan, and identifying possible bottlenecks in either the infrastructure or process.

Because users often require trade-offs between speed and accuracy, the model has been designed so that different degrees of detail can be applied to different portions of the scenario data. Thus, the experienced analyst can focus on specific aspects of the plan when time is limited. These detail/aggregation choices are discussed later.

Six modes of transportation are modeled in ELIST: road (including line haul operations), rail, water, air

equipment (they had moved via different modes to the theater), and performing final training activities for personnel before a mission. The units then move to their tactical positions via various modes of transportation either specified in the requirements or chosen by the simulation. Finally, the units merge into an integrated force. Other optional activities can also occur, such as follow-on movements and delivery of sustainment.

2.2 Vehicle Characteristics

ELIST8 contains data on every vehicle type that could be used. The model uses information, such as payload, cargo area dimensions, curb weight, average speed, and average on- and offload times. ELIST8 uses a vehicle grouping called an *asset* to specify preferences on how to move specific commodities. An asset is a set of vehicle types that will be used in the same way in the plan. Examples are a set of heavy trucks or a set of tanker railcars.

Commodity asset rules list (in order of preference for each commodity) the assets that can be used to transport that commodity. Conditions based on lateness and distance can be included in these rules. A specific collection of vehicles is defined.

2.3 Network Infrastructure

A theater infrastructure is required for the area of operations and must include all the nodes (i.e., ports, staging areas, and destinations) referenced in the ETPFDD. Links — roads, railways, waterways, and pipelines — connect the nodes. All nodes have geographic coordinates for mapping and distance calculations. ELIST8 models infrastructure resources primarily in four ways: rate resources, gate resources, capacity resources, or discrete resources. A rate resource is characterized by its ability to process a given amount of cargo in a given time, for example, the capability to load containers on railcars in containers/hour. The resource represents the personnel and equipment involved. The time consumed by an activity using a rate resource will be at least as long as the amount that is being processed divided by the rate. The activity, however, can take longer than the time dictated by one resource, for example, when multiple resources with different rate values are required.

A gate resource also represents a capability specified as a rate. This type of resource, however, does not have a duration associated with its use. Road capacities, for example, are represented using gate resources; a number of vehicles may travel onto the road per hour.

A capacity resource is a nonconsumable resource that is used in some fraction of the whole for the duration of an activity. Its use, however, does not affect the duration of the activity. An example is storage area at a port. Some

equipment is represented discretely. Cranes at a berth, for example, are acquired for a ship-loading activity and released when the activity is complete.

The main nodes modeled are seaports, airports, and intersections. The basic node, an intersection, can represent a staging area, a rail terminal, or any type of intersection of links. Trucks, railcars, and helicopters can be on- and offloaded at an intersection. Capabilities for loading cargo to/from vehicles are modeled as rate resources. These capabilities are broken into infrastructure, materials handling equipment, and personnel, giving analysts flexibility for storing the static aspects of a terminal, such as ramp and dock capability, separately from more dynamic capabilities, such as personnel and equipment. From these values, a rate resource is constructed for the simulation, which has a capability that is the minimum of the three components.

Seaports have all the data associated with an intersection node, plus the data for ship berthing and loading.

Aircraft parking areas at airports are explicitly modeled. Only a limited number of aircraft can be processed at a time because of restrictions on tarmac areas. All aircraft-loading resources can operate anywhere within the airport.

The primary attributes of infrastructure links used by the simulation are length, rate of march, and rate of entry. Limited link capacity is modeled by the rate at which vehicles can start onto the link (rate of entry). Road and rail links can also be limited by the size and weight of vehicles that can traverse them.

2.4 Scenario Parameters

A scenario is a specific run of the model. The analyst can specify a number of options and parameters to direct and tune the simulation for each scenario. A few examples are given. The simulation can either enforce the mode of travel specified in the ETPFDD or select a mode on the basis of the situation. The maximum time to wait for additional cargo and the percentage of capacity preferred prior to departure can be specified for various vehicle types. The analyst can decide whether the storage area at a node should constrain movement or the cargo should be allowed simply to overflow. The preferred number of railcars in a train can be input, and the analyst can decide whether trucks must travel in serials (groups of vehicles). Scenario parameters are stored in the database as are all other ELIST8 data.

3. SIMULATION

The simulation is driven by discrete events; for a detailed discussion, see [3]. Initially, scheduled events are unit

availability, ship arrivals, and plane arrivals. All of these events have participant lists that contain unit component entities. Whenever an activity is completed, the newly idle unit components look in their required activity lists and create and schedule new activities as appropriate.

Unit components keep detailed histories of each completed activity as well as time spent queuing. This information is available to the analyst. An example history is provided below for a piece of equipment as it moves through the activities required in the simulation (Table 1). For each time, an activity provides information on what actions occurred at a specific location.

Table 1 Example History of Equipment Movement

Day/Time	Activity
54 23:03	Strategic Ship Arrival and Berthing (SAFAQIS -VKNP-PRT)
55 2:03	Off-load from Advantage Ship at Berth CONT-2
55 3:00	Any Mode Trip from SAFAQIS -VKNP PRT to TEBESSA -WSVC-CAP
55 3:00	Train On-loading (Railcars)
55 3:00	Waiting for Train for SAFAQIS-VKNP-PRT to TEBESSA -WSVC-CAP (Rail)
55 7:00	Train Departure (Railcars)
55 17:15	Train Arrival (Railcars) TEBESSA -WSVC-CAP
55 17:15	Train Enter Node (Railcars)
55 17:44	Train Off-loading (Railcars)
25 6:45	Waiting for Marry-Up
61 2:45	Marry-up (Equipment and Personnel)
63 2:45	Road trip from TEBESSA -WSVC-CAP to BIR EL ATER -BQWK-APT
63 2:45	Waiting for Serial for TEBESSA WSVC-CAP to BIR EL ATER -BQWK-APT (Convoy)
63 2:45	Road (Self: 1.0 vehicles) Departure in Convoy
63 5:46	Road (Self: 1.0 vehicles) Arrival BIR EL ATER -BQWK-APT
63 5:46	Road (Self: 1.0 vehicles) Enter Node BIR EL ATER -BQWK-APT
63 6:15	Final Delivery BIR EL ATER -BQWK-APT

3.1 Activity Objects

Activities are implemented as objects, each with an “execute” method. Some activities can be considered as compound activities and are represented as a single object with multiple states. A trip with an asset, for example, actually consists of a fixed series of subactivities — cargo onload, travel, node entry or parking, and cargo offload — with the asset, route, and participant data remaining the same throughout.

The subclass capability and interface construct of Java were very effective in implementing the different activities of the simulation, primarily because of the many similarities among the transport activities for different modes, different cargo types, trips with or without cargo, and trips with or without assets.

3.2 Event and Asset Managers

An event manager object handles the scheduling and execution of events. Events generate priorities on the basis of the delivery times required by participants. The simulation can either be run to a specific time or be run and stopped as desired.

An asset manager object oversees the allocation of assets to activities. Vehicles are chosen on the basis of commodity-asset preference rules, specific cargo dimensions, current location of the cargo and the vehicle (if fully tracked), and service area of the vehicle’s asset pool. When an asset vehicle completes a trip, and there is a queue for the asset, it is either assigned immediately, if needed at its current location, or sent by the asset manager to the location of the highest priority event in the queue. If no queue exists, the asset is sent to its home node.

3.3 Results

Many reporting options are available to assist the analyst in evaluating the success of a plan, identifying bottlenecks, and analyzing asset usage. The success of the plan depends on the on-time arrival of all units at their final destinations.

Summary arrival results can be viewed in total short-tons for the entire scenario or by cargo type or destination. Figure 3 shows a sample delivery graph. The graph areas plot the arrival of cargo through the various locations in the theater. These can be compared with the requirement

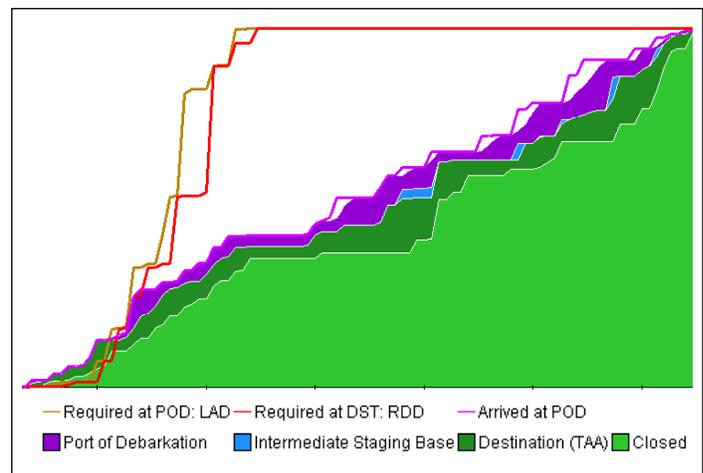


Figure 3 Sample Delivery Graph of Short Tons vs Time

lines. When the delivery is above the requirement, the cargo is early; when it is below the requirement, the cargo is late.

A query facility allows specification of groups of units on the basis of name, location, delivery, and many other factors. These groups or individuals can then be reported and graphed. For example, all units traveling to a certain destination or all units not yet arrived could be reported.

Usage-over-time data are saved for all network and asset resources and all queues. The user interface allows easy selection (for reporting) from among all queues, destinations, seaports and airports, road links, routes, assets, asset pools, etc.

4. SYSTEM ARCHITECTURE

Because ELIST8 is a complex system, various conceptual levels have been designed to manage this complexity (Figure 4). First, all object representations have been separated from the user interface and persistent representations. User interface items are applied consistently in their own layers, making each model object cleaner and more maintainable. For example, if requirements called for changing the interface, the underlying representation would not be affected, nor would it affect the simulation code or the interfaces with other external models. In ELIST8, the persistence mechanisms were moved into a separate layer to reduce the redundancy of database calls in each object and to hide the implementation of saving the data from each object. As a result, each object has a very thin persistence layer, mapping its attributes to relational tables. This capability not only increases productivity in development, but also significantly enhances system maintainability.

Finally, this layer now gives the option to port to other databases (relational or object based) with limited impact on base objects.

All interactions with external models are encapsulated within their own layer, allowing easy plug-and-play functionality. For example, High Level Architecture (HLA) has been integrated into ELIST8 in a separate layer, allowing the model to run either inside the Analysis of Mobility Platform HLA Federation or as a stand-alone system.

5. DEVELOPMENT OF ELIST IN JAVA

The new version of ELIST was developed in Java for several reasons. Java offers the advantages of an object oriented programming language. Java provides the platform independence required and native support for a number of required and potentially useful capabilities. The JDBC (Java Database Connectivity) toolkit supports multiple DBMSs. Graphical user interface, networking and Internet, and multi-threading capabilities are all available within the base Java, allowing the team to exploit new technologies as required. Two disadvantages, however, were identified. First, the language was “immature,” and changes in Java’s ongoing development might affect Argonne’s effort. Second, the execution speed of this semi-interpreted language might hamper acceptance of the system, although it was assumed that the execution speed would be resolved in time.

The overall feeling of the team toward Java has been positive, but there have been problems. No single environment satisfied everyone. The team used several environments, including VisualAge for Java (IBM), Forte (Sun Microsystems), and JBuilder (free, Foundation

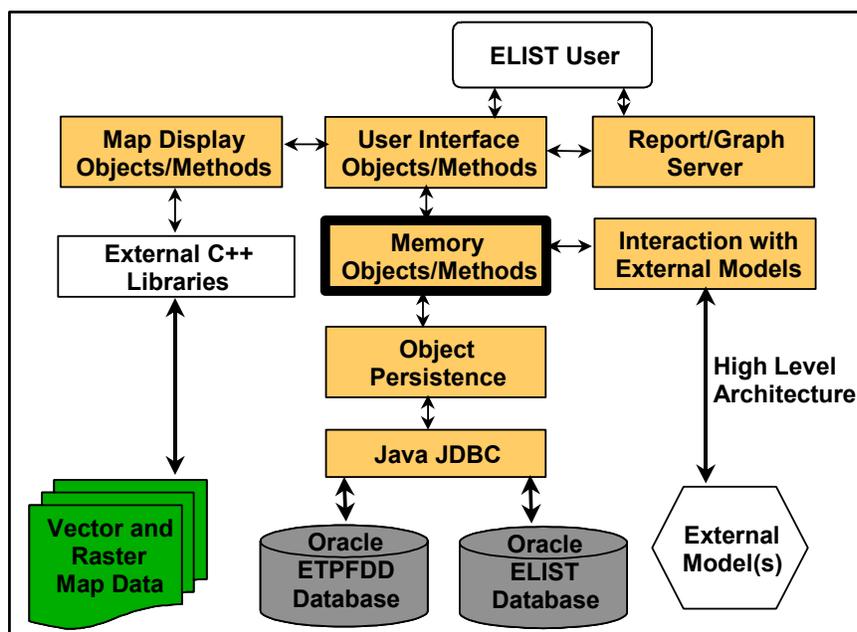


Figure 4 System Architecture

version from Borland/Inprise), or external editors in conjunction with Sun's Java (virtual machine) and jdb (debugger).

The immaturity of Java required the team to spend time on "work-a-rounds," especially in relation to the Swing user interface classes. The Java Developer Connection site (<http://developer.java.sun.com/index.html>) (Bug Parade) was beneficial in identifying Java problems and solutions, and the work-a-rounds did not consume a significant amount of development time. These problems, which were most prevalent during the first couple of years of development, have decreased significantly in recent years.

The speed of execution and memory constraints continued to be a concern. The team has been very careful in limiting how memory is used because Java has a fixed heap size of 1.5 GB. The deployment of JDK1.3 and the Just In Time (JIT) compiler greatly alleviated the speed concerns. As an illustration, a relatively large scenario that consists of 20,000 units (approximately 300,000 personnel and 3 million short tons of equipment and supplies) using three modes (road, rail, and water) was simulated in slightly more than 30 minutes on a Sun Ultra 10 workstation.

6. CONCLUSION

The ELIST8 simulation is a critical piece of an extensive system that includes import, export, visual editing, and error checking of vehicle data, asset and commodity data, network data, ETPFDD data, and scenario parameters. Mapping and query capabilities are included and continue to be enhanced. The capabilities provided by this expanding collection of tools assist military logistics personnel with both data management and analysis tasks. By providing greater ease and speed for modifying and analyzing the plans, these tools enable examination of more "what if" scenarios and facilitate the logistical planning process.

Overall, Java was found to be scalable, allowing not only complex class/object interactions but scaling to tens of thousands of objects. Since the distribution of the JIT, concerns regarding the speed of execution have disappeared.

7. ACKNOWLEDGMENT

This work was supported under a military interdepartmental purchase request from the U.S. Department of Defense, Military Traffic Management Command Transportation Engineering Agency (MTMCTEA), through the U.S. Department of Energy contract W-31-109-ENG-109.

REFERENCES

- [1] C. Macal, C. Van Groningen, and M. Braun, Simulation of transportation movements over constrained infrastructure networks, *Proc. 1995 Simulation Multi-Conference*, Phoenix, AZ, (4):97-102, April 27, 1995.
- [2] M.D. Braun and C.N. Van Groningen, *ELIST 8 transportation model*, ANL/DIS/02-1, Argonne National Laboratory, Argonne, IL, Feb. 2002.
- [3] M. Braun, G. Lurie, K. Simunich, C. Van Groningen, H. Vander Zee, and M. Widing, ELIST8: A simulation system for transportation logistics planning support, *Proc. 2000 Summer Computer Simulation Conference*, Vancouver, BC, July 16-20, 2000.