Simulator for Port Container Terminal Using An Object-Oriented Approach

Yong-Seok CHOI, Tae-Young HA

Korea Maritime Institute

#1027-4, Bangbae 3-dong, Seocho-gu, Seoul, 137-851, Korea

E-mail: drasto@kmi.re.kr; haty@kmi.re.kr

Key words: container terminal, object-oriented approach, simulation

ABSTRACT

Since world container throughput continues to grow, the main issues facing decision-makers at port container terminals are how to expand the existing container terminals and construct new container terminals. Simulations can be used in the decision making process tools due to its ease and ability to reflect the real world system. The object-oriented approach provides for both reusability and modularity that best fits these requirements. In this paper, the object-oriented approach to modeling and simulating general container terminals is presented. Simulation tools based on Visual C++ provide a user-friendly input and output environment through the use of an object class-library. This paper also presents the case study of a simulation of a real container terminal in Pusan, Korea.

1. Introduction

The most difficult problems in port container terminals are whether the existing container terminals are efficient enough to handle large container streams, or whether the various operation systems using transfer cranes and container cranes are effective. In order to investigate the efficiency and effectiveness of container terminals, port container terminal simulators have been developed.

Simulation has been widely applied to port container terminals all over the world because it efficiently analyzes system performances. Simulation is perhaps the best tool to study any non-trivial, real world systems, in genera , and an excellent approach by which to quantify container logistics, in specific.

For analysis of complex and large systems, simulation is often used prior to operating the real world system to mediate the dynamic situation [5]. Therefore simulation has been recommended to analyze port container terminal systems. Park & Noh used a Monte Carlo type simulation approach to plan port capacity using SLAM [8]. Lai & Lam examined strategies for the allocation of container yard equipment for a large container yard in Hong Kong [4]. Ramani provided a justification for modeling port operations through simulation rather than through analytical queueing models [9]. Hayuth used a discrete event simulation to build a port simulator [3]. But those simulation models or simulators set limits to the particular container terminal based upon port properties and are not sufficient to analyze the operations of a dedicated modern container terminal which are equipped with newly sophisticated container handling equipment.

Furthermore, recent research has used an object-oriented simulation based upon simulation software such as MODSIM, GPSS/H, ARENA, and Proof [2] [6] [7] [10] [12]. Therefore, the trends of simulator development are as follows:

1) User-friendly environments with various statistical output,
2) Animation functions to provide verification and validation, and
3) Object-oriented approach to provide reusability and extensibility [11]

To successfully design and develop a port container terminal, requires a simulation tool that supports rapid modeling and simulation and that is easy to use and sufficiently complex and flexible to model real container terminal systems. It is therefore necessary to consider above three trends at the same time.

The objective of this study is to develop a port container terminal simulation tool using an object-oriented approach in order to provide user flexibility. The research issue is categorized following three concepts. The first concept is the development of simulation models to describe the container logistics for a general container terminal. This model was developed using an object-oriented approach and was designed with an object class

library. The second concept is the development of output statistics to measure the performance of resources. To get detail statistics we use the number of state transitions and the cumulative times over facilities and equipment. The third concept is to use 2-D animation to validate the operation of a port container terminal and to provide a user-friendly environment. The results of the simulation are analyzed using the output statistics and the animation during the simulation run.
The simulator is developed using C++ language based on object-oriented programming and Microsoft Access as database system. Two cases are considered and we test and analyze its performance to validate the developed simulator.

2. Object-Oriented Simulation

2.1 Object-Oriented Approach

The object-oriented approach is popular for programming and modeling. It supports data abstraction, encapsulation (hiding information), inheritance, and dynamic binding to enable programmers and analysts to create complex models by reusing models created earlier [1]. Data abstraction and encapsulation are tools used in the object-oriented approach to provide modularity. Inheritance and dynamic binding are simultaneously used to provide reusability. Modularity and reusability makes an object-oriented simulation the best candidate for simulating container terminal functions.
For modeling, each component of equipment is created as a class with properties analogous to their real-world counterparts. The use of objects in a set of classes provides important benefits in the way of features for managing complexity. The ability to send and receive messages between objects is essential for effective communication. This allows for object autonomy by only permitting objects to affect each other via messages.

2.2 Object Modeling

Fig. 1 describes an object scheme that represents the relationship between classes needed to model a container terminal. The relationships include 1) one and only one, 2) one or many, 3) from zero to many, and 4) from one to many. The object scheme presented in Fig. 1 stands for relationship between the material flow objects in terms of the container handling operation.

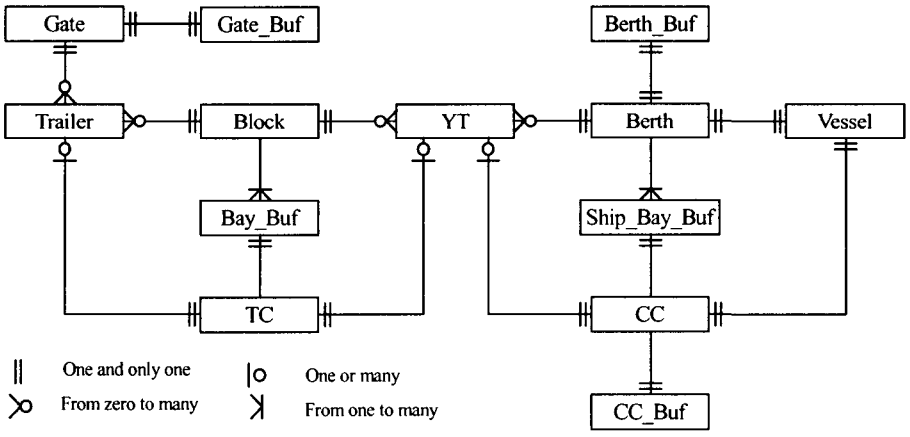

Fig. 1. Object scheme in container terminal

The key facilities and equipments are defined as the various types of objects. The equipments such as transfer cranes (TC), container cranes (CC), yard tractors (YT), vessels, and trailers are defined as movable objects and the facilities including gate, yard, and berth as stationary objects, which is occupied by movable objects. The buffer makes connections between facility and equipment. We defined Gate_Buf as the line of trailers waiting in front of the gate, Bay_Buf as the line of waiting trailers and YTs, Berth_Buf as the line of waiting vessels, and Ship_Bay_Buf as the line of waiting YTs.
In the container terminal, there are two major flows, the material flow and information flow. Materials (e.g. container) move forward through the material handling equipments and transporters. Information, however, flows backward through the container terminal by user-defined information (attributes and methods). Since the

behaviors of containers and user-defined information are different, the information tables for container handling such as a work list, a sequence list, and storage information are needed in defining objects.
Each attribute and method in the object scheme represents an object. The five major classes used in the object scheme are EQUIPMENT, TRANSPORTER, FACILITY, CONTAINER, and BUFFER as defined in Table 1. The analogue of BUFFER is queue, of CONTAINER entity, of EQUIPMENT, TRANSPORTER, and FACILITY resources in most simulation systems. The attribute section of an object describes the dynamic and static status and the results of decision-making. An attribute is data that each object in a class has its own value. The basic attributes are programmed into the simulator system in terms of super class. The basic attributes of equipment are object id, object name, speed, and process time and the basic methods are 'move' and 'work'.
For simulation we add the attributes related with equipment operation. The basic attributes of a transporter are object id and ship type. In the case of facility, the basic attributes are object id and object name. In Table 1, for example, a container needs an attribute 'in_out' to indicate whether it is an import container or an export container, and a 'destination' attribute to point out to which location the container transports.

Table 1 Attributes and methods of classes

Super Class	Class	Attributes	Methods
EQUIPMENT	CC	work_state, work_type, current_location destination, *sequence_list	move(), decision_dest() work(), work_control()
	TC	work_state, work_type, current_location destination, *work_list	move(), work(), work_control()
TRANSPORTER	YT	yard_in, work_state, speed, destination	move(), work_control()
	Trailer	in_out, state, speed, destination	move(), work_control()
	Vessel	speed, arrival_time, departure_time, export_num, import_num, trans_num	-
FACILITY	Gate	type, service_time	decision_dest(), process()
	Block	type, specification, *block_info	tc_position()
	Berth	ship_id, assigned_block, occupied_mode	create_cc_sequence_list()
CONTAINER	Container	in_out, size, state, destination	-
BUFFER	Gate_Buf	location, capacity	enqueue(), dequeue()
	Bay_Buf	location, capacity	enqueue(), dequeue()
	Berth_Buf	location, capacity	enqueue(), dequeue()
	CC_Buf	location, capacity	enqueue(), dequeue()
	Ship_Bay_Buf	location, capacity	enqueue(), dequeue()

The method section contains procedures and knowledge needed to make decisions for operative control and to send messages to the other objects. The 'move()' method of CC and TC is defined in equipment, and it controls the movement from current location to destination.

2.3 Simulation Mechanism

The simulator consists of two main areas to support the object-oriented simulation. Fig. 2 shows the first as a system area that controls the events and the second as a simulation model area which simulates the container terminal.
In the system area, the event controller is the program module for processing events as they flow through the system. Events are generated and put onto the event list in relation to the simulation clock. The event list processes events on the basis of the time associated with the request; in addition, it allows events to be interrupted and rescheduled when higher-priority events must be processed.
In the simulation model area, each object has its own state transition network. During the simulation run the event controller will take the first pending event off the event list and instruct the object to update itself. The object performs its own time advance and state changes via its state transition network. The states of movable objects consist of 'work', 'move', 'wait', and 'idle', these states represent the operational situation of equipment and vehicles. In stationary objects, the state is either 'occupied' or 'unoccupied'.
The architecture can therefore support modeling of objects ranging from a simple buffer to an object representing a container handling, such as the TC and CC, as well as revising displays and updating results at user-defined times.

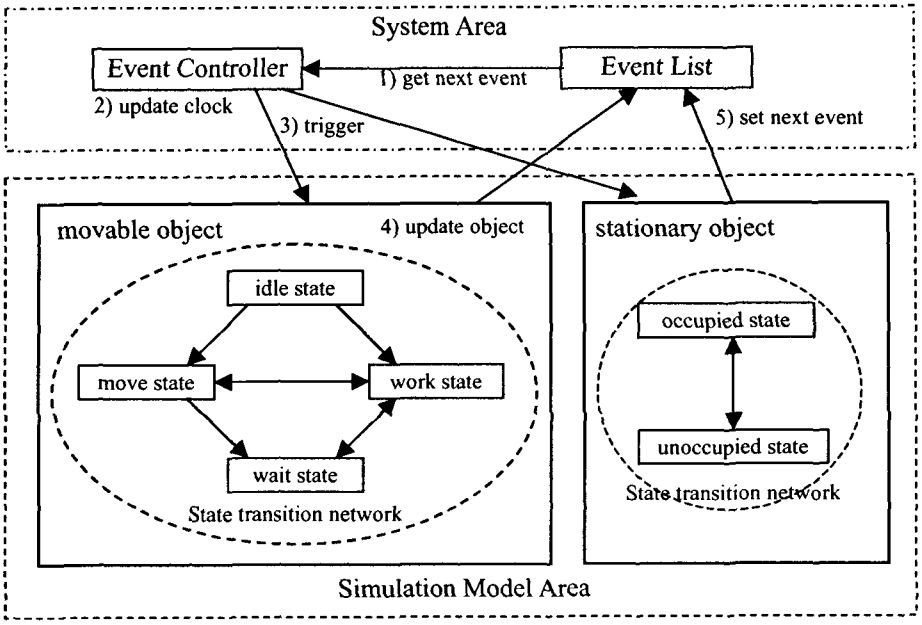

Fig. 2. Simulation mechanism

The event list is designed to control the simulation schedule and contains 'Object_Name', 'Object_Id', 'Event_id', and 'Event_Time' for events (see Table 2); hence any object that conforms to a given standard can be placed on the event list and be instructed to update itself. The event controller retrieves the earliest event from the event list and triggers the next event.

Table 2 Event list to control the simulation event

Object_Name	Object_Id	Event_Id	Event_Time	Description
Yt	31	18	864004	Yt #31 moves at time 864004 sec.
Yt	20	18	864013	Yt #20 moves at time 864004 sec.
Tc	16	23	864016	Tc #16 arrives in the destination at time 864016 sec.
Tr	6	6	864019	Trailer #6 arrives in the destination at time 864019 sec.
I20	26	1	864034	Trailer with 20 feet container generates at time 864034 sec.
NextShip	3	41	868352	Next ship arrives at time 868352 sec.

3. Simulation System

The simulation system is programmed in the general-purpose language, Visual C++, based on object-oriented programming. Object-oriented modeling methodology for object-oriented simulation is used. This approach allows models to be built and modified easily. The developed system along with the associated modules is shown schematically in Fig. 3.

Fig. 3 describes the general flow of control in our simulator system. Through the user interface user inputs the parameters to construct a container terminal within the simulator. Based on these parameters, the system automatically constructs an initial model. Then user can modify this initial model. The modified model is simulated using an object-oriented simulation, and its performance recorded in a database. The system can be used either for initial design of facilities or to experiment with different operating policies (e.g. compare the performance of competing alternative design and policies).

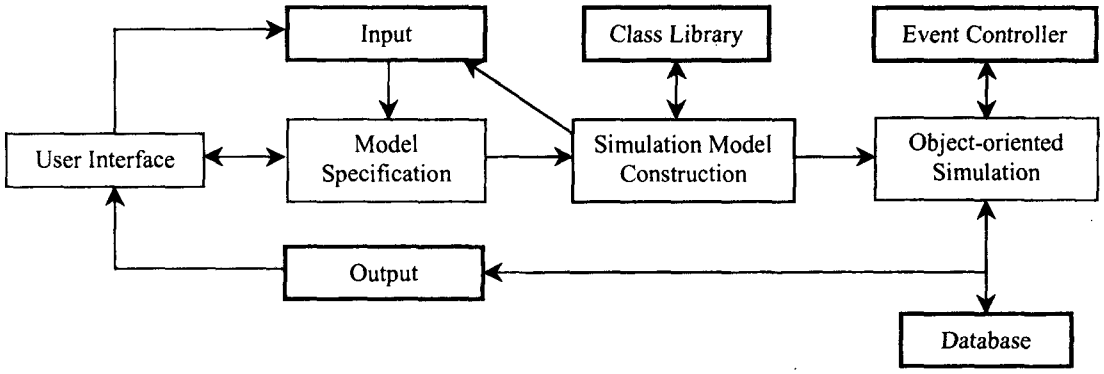

Fig. 3. Simulator structure

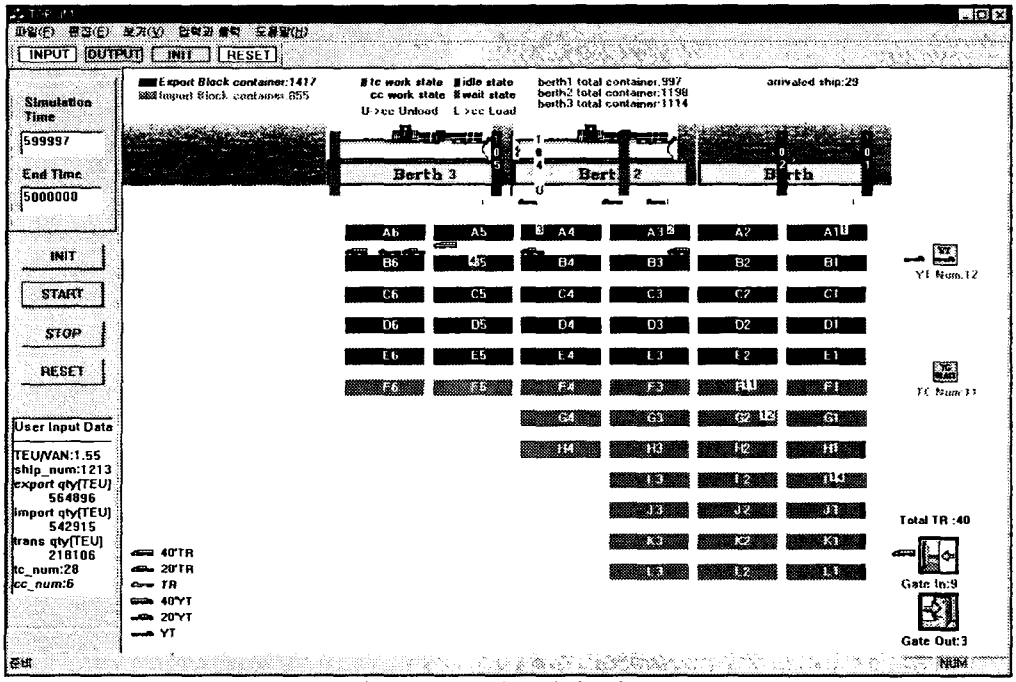

Fig. 4. Developed simulator

The on-chassis system, the straddle carrier system, and the transfer crane system are ones of the most popular handling systems in the container terminals. This classification in mainly based on the yard-side equipment. Two typical types of yard-side equipment are the straddle carrier and the transfer crane. Currently, the transfer crane system is more common type in the container terminals in Korea. Therefore, this system deals with the transfer crane system as the yard-side equipment and illustrates operations on container terminal with typical transfer crane system.

Fig. 4 shows the configuration of the experiment model as displayed by the developed simulator. Two vessels come alongside the berth 2 and 3 and are being unloaded by four CCs (currently, only CC104 is active, though, as indicated by the light color). Containers are to be positioned in blocks of yard area. In these yard areas, the seven TCs are working.

3.1 User Interface

The graphical user interface consists of windows and dialog boxes that support (1) construction and modification of the model, (2) setup of operational parameters, (3) simulation and animation, and (4) analysis of output graphs and reports. The user interface has been developed to facilitate entry and manipulation of scenarios, as well as to present results to users in a logical manner.

3.2 Class Library

Classes are categorized as abstract, support, and application. Abstract classes provide the basic properties, such as speed, and the state that the application classes inherit. Support classes include the event list, simulation clock, and distribution generators. Application classes contain all of the behaviors specific to the class; for example, the crane class has behaviors such as loading, unloading, moving, and waiting.

3.3 Input Module

The inputs to the simulator consist of (1) container throughput data, (2) operational policies, (3) equipment characteristics, and (4) facility requirements. These input data are needed to construct the model system. Fig. 5 illustrates the input dialog box for annual throughput data. Fig. 6 illustrates the input dialog box for equipment characteristics.

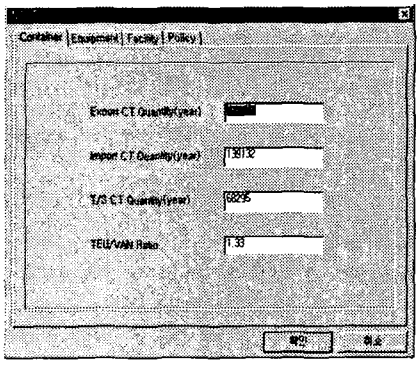

Fig. 5. Container dialog

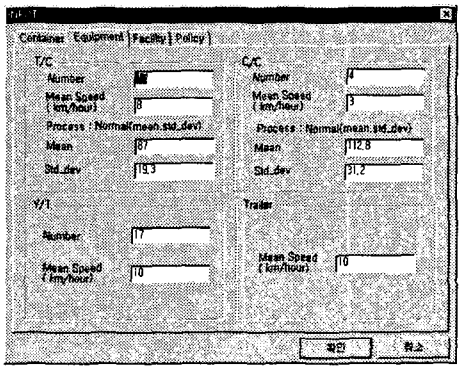

Fig. 6. Equipment dialog

Fig. 7 illustrates the input dialog box for facilities. Fig. 8 illustrates the input dialog box for the operation policies of the model systems. The operation policies play an important role in the construction of operation logic.

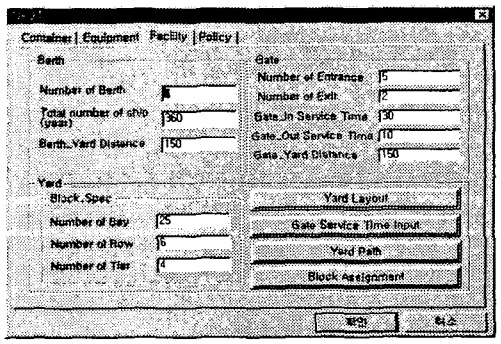

Fig. 7. Facility dialog

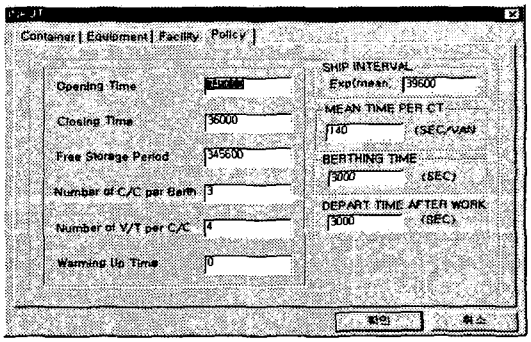

Fig. 8. Policy dialog

3.4 Output Module

The output module provides a high-level view of the throughput capability of the container terminal, as well as how efficiently its resources are used. These outputs allow analysts to compare the simulation scenario with alternate scenarios and identify the facilities within the container terminal that are bottlenecks. In addition, analysts can determine which resources are not being used efficiently. Fig. 9 illustrates one of the resource utilization windows (e.g., transfer crane statistics). Fig. 10 illustrates the vessel statistics window. Other resources for which utilization windows are available include (1) gates, (2) yards, (3) berths, (4) container cranes, (5) transfer cranes, (6) yard tractors, (7) trailers, and (8) ships. The statistics are displayed for all objects of a class during any a given period. The graphs are created automatically in the format of statistics.

The outputs are generated and collected during each simulation run by each of the simulation objects. For example, CC collects results on the time rates 'work', 'move', 'wait', and 'idle'. The statistics are collected according to the user-defined warm-up period and period of record.

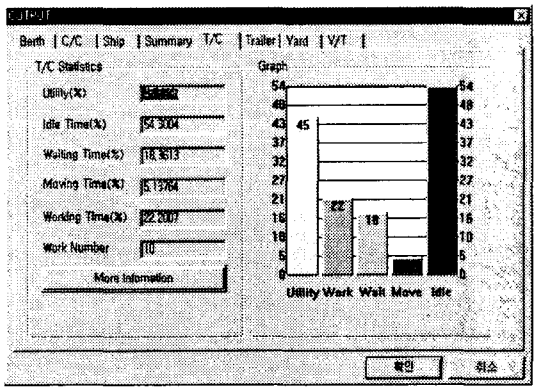

Fig. 9. Statistics window of transfer crane

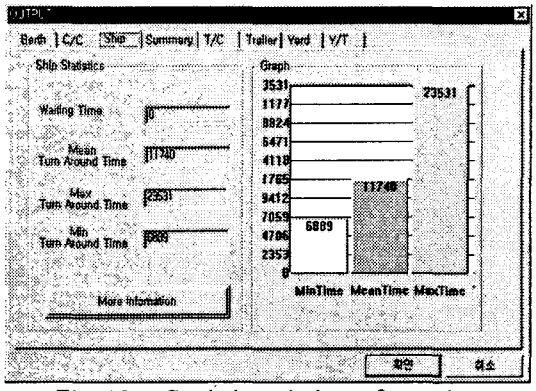

Fig. 10. Statistics window of vessel

4. Simulation and Analysis

The simulator is used as a testbed to assist management in comparing the results of computer- generated resource allocation policies with their own experience.
To validate the developed simulator, we used two real systems, UTC (Uam Terminal Co.) and PECT (Pusan East Container Terminal) located in Pusan, Korea as the model systems in this case study.

4.1 Experiment Design

These models consider the values of the various parameters of facility operations and the same criteria are used to evaluate the system effectiveness of the developed simulator. The parameters are summarized in Table 3 and consist of annual container throughput and the facilities and equipment requirements.

Table 3 Historical data for two cases

Items	Case 1 (UTC)	Case 2 (PECT)
Annual export container throughput(TEU)	150,515	564,896
Annual import container throughput(TEU)	139,132	542,915
Annual transshipment container throughput(TEU)	68,295	218,106
Ratio of TEU/VAN	1.33	1.55
Annual total number of handled vessels	360	1213
Number of berths	2	3
Number of container cranes	4	6
Number of transfer cranes	10	28
Number of yard tractors	16	24

4.2 Input Data

The two model systems use the TC system as yard side equipment and the same operation flows. Therefore, as operation policies, the same parameters as input in the two cases are used.
Opening time of receiving containers before vessel's arrival schedule : four days
Closing time of receiving containers before vessel's arrival schedule : 10 hours
Free storage periods for the imported (inbound) containers : four days
The number of allocated CC per berth : two
The number of allocated YT per CC : four

Table 4 Equipment characteristics

Characteristics \ Equipment		CC	TC	YT	Trailer
Speed (km/h)		2.7	8.04	10	10
Operation time (min.)		N(112.8, 31.2)	N(87, 193)	-	-
Number of equipment	UTC	4	10	16	-
	PECT	6	28	24	-

Other input data consist of container throughput data, equipment characteristics, and facility requirements. We use the annual throughput data in Table 3 as the container throughput data. The equipment characteristics are summarized in Table 4. Table 5 shows the facility requirements.

Table 5 Facility requirements

Item / Model	Gate			Yard		
	Entrance	Exit	Service time(sec.)	Export block	Import block	Block capacity
UTC	2	1	UN(20,30)	7	5	25bays*4tiers*6rows
PECT	9	3	UN(20,30)	30	26	25bays*4tiers*6rows

4.3 Experiment and Results

Before the results of the simulation runs are collected, the warm-up period is set at ten days. The length of each simulation run is set at 20 days. The output statistics are obtained from a sample of five independent simulation runs.

4.3.1 Results of Case 1

The simulation results of the Case 1 (UTC) are shown in Tables 6 and 7. Table 6 represents the simulation results of berth and yard activities, including the occupancy rates of berth and yard, the mean handling time of a container while in berth, as well as the storage level of blocks. The berth statistics reflect the level of demand for berth service through the occupancy rate. It is defined as the percentage of the occupied time during the simulation period. In Table 6, the average occupancy rate of yard including export blocks and import blocks is approximately 40%. This means that the average number of containers stored in a block is 245.

Table 6 Output of berth and yard in Case 1

Berth			Yard		
	Occupancy rate(%)	Mean handling time(sec.)		Occupancy rate(%)	Storage VAN
Berth 1	87.0	182	Export blocks	43.68	261
Berth 2	86.0	216	Import blocks	37.11	222
Average	86.5	199	Average	40.40	245

Table 7 represents the simulation results of equipment. Since CC, TC, and YT are the resources of the terminal, utilization statistics of these equipments are needed. To acquire more detail statistics we separate the four other rates from the utilization rate. These time rates are calculated using the state transition network of objects. In the case of YT, turnaround time is the cycle time from one CC operation to the next CC operation and waiting time. The berthing time for each vessel and the service time for each trailer are the total time taken by the facility.

Table 7 Output of equipment in Case 1

Resources	Performance Measures				
	Utilization(%)	Work time(%)	Move time(%)	Wait time(%)	Idle time(%)
CC	84.4	45.5	0.79	34.0	19.0
TC	45.7	22.2	5.14	18.4	54.3
	Utilization(%)	Turnaround time(sec.)	TC waiting time(sec.)	CC waiting time(sec.)	
YT	65.3	652.95	58.97	34.30	
	Min berthing time(hr.)	Mean berthing time(hr.)	Max berthing time(hr.)		
Vessel	13.98	24.3	46.45		
	Min service time(min.)	Mean service time(min.)	Max service time(min.)		
Trailer	2.63	12.03	201.37		

Table 8 Output of berth and yard in Case 2

Berth			Yard		
	Occupancy rate(%)	Mean handling time(sec.)		Occupancy rate(%)	Storage VAN
Berth 1	86.0	134	Export blocks	27.06	156
Berth 2	87.0	136	Import blocks	29.34	176
Berth 3	78.0	125			
Average	83.7	131.66	Average	27.70	166

4.3.2 Results of Case 2

The simulation results of Case 2 (PECT) are presented in Tables 8 and 9. In Table 8, since PECT has three berths, the utilization rate of berth 3 was estimated low under the average occupancy and the mean handling time per container. The occupancy rate of yard usage was estimated low as compared with Case 1 (UTC). This result tells us that the yard size of PECT may be enough to handle the current throughput.

Table 9 shows the simulation results of equipment. Though the utilization of CC is estimated to average 81.8%, the average utilization of TC is estimated to average 31.1%. This low utilization is due to high idle times. In other words, PECT possessed comparatively more TCs for handling the given container throughput.

Table 9 Output of equipment in case 2

Resources	Performance Measures				
	Utilization(%)	Work time(%)	Move time(%)	Wait time(%)	Idle time(%)
CC	81.8	64.8	0.3	16.7	12.0
TC	31.1	18.5	6.6	5.9	68.9
	Utilization(%)	Turnaround time(sec.)	TC waiting time(sec.)	CC waiting time(sec.)	
YT	54.8	389.24	134.29	70.68	
	Min berthing time(hr.)	Mean berthing time(hr.)	Max berthing time(hr.)		
Vessel	6.7	21.2	40.5		
	Min service time(min.)	Mean service time(min.)	Max service time(min.)		
Trailer	2.65	10.78	122		

5. System Validation

After developing our simulator, we try to evaluate and validate the developed simulator with real data concerning two case models. First, we compare its behavior with that of the real terminal. In particular, we compare the total operation time to complete all work required during a given time period with the simulation time required in the simulator to complete the same operations. Results show that our model is close to the real terminal behavior in terms of the average statistics.

The most important issue in developing a simulator is to verify whether this system could adequately represent the container terminal operation. Since both the container terminal operation and the simulator are stochastic, the statistical properties of the true system's output should be compatible with those of the simulated outputs. In order to validate the developed system, we used the 1996 annual throughput data as historical data. The historical data of the Case 1 and the Case 2 were compared to simulated results in the following categories:

(1) Facility Performances: Berth occupancy, Yard occupancy, and

(2) Equipment Performances: CC utilization, TC utilization, YT utilization.

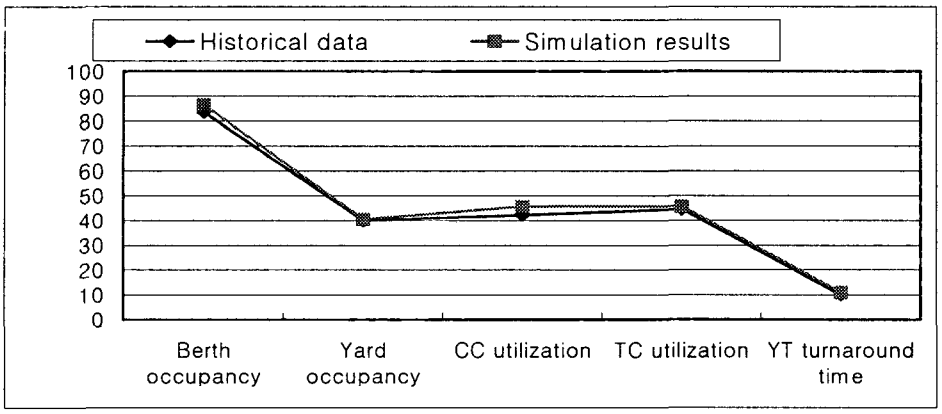

Fig. 11. Case 1 results

In Fig. 11, the simulation results are similar to those of UTC, except that CC utilization rate is overestimated. However, in Fig. 12, the simulator may underestimate TC utilization rates. It has been later found that the PECT did not use all available TCs, because some of the TCs were not assigned. As the TC utilization rate decreases, YT utilization rate increases. Therefore, we also find that the simulator overestimated YT utilization rate.

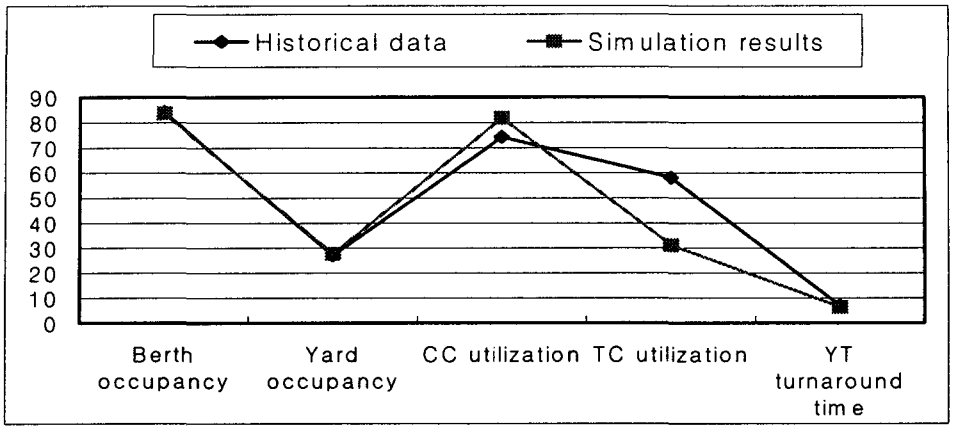

Fig. 12. Case 2 results

As shown in Fig. 11 and Fig. 12, the developed simulator gave accurate statistics at different points of facility performances

6. Conclusions

In this study, a simulator for port container terminal systems based on an object-oriented approach is developed. The developed simulator provides a user friendly input and output environment through the use of an object class-library. And the presented object modeling and simulation mechanism may also lead to simulation design used in developing the simulation system. Based these assumption, the proposed simulation methodology is implemented through the use of Visual C++.

To validate the developed simulator we performed the simulation experiment on two real container terminals, UTC and PECT. The results of the simulation experiment were analyzed using the output statistics. The output of resource statistics was acquired through the state transition network during the simulation run.

These resource statistics could be used for the capacity analysis and the operational efficiency analysis of existing container terminal.

References

(1) David R.C.H. (1996): Object-Oriented Analysis and Simulation, Addison-Wesley Publishing Company

(2) Gambardella LM, Rizzoli AE, Zaffalon M (1998): Simulation and Planning of Intermodal Container Terminal, *Simulation* 71(2), 107-116

(3) Hayuth Y, Pollatsch M.A., Roll, Y. (1994): Building a Port Simulator, *Simulation* 63(3), 179-189

(4) Lai K, Lam K (1994): A Study of Container Yard Equipment Allocation Strategy in Hong Kong, *International Journal of Modeling and Simulation* 14(3), 134-138

(5) Law A, Kelton W (1991): Simulation Modeling and Analysis, 2nd edn. McGraw-Hill New York

(6) Nevins M.R., Macal C.M., Joines J.C. (1998): A Discrete-Event Simulation Model for Seaport Operations, *Simulation* 70(4), 213-223

(7) Nevins M.R., Macal C.M., Love R.J., Bragen M.J. (1998): Simulation, Animation and Visualization of Seaport Operations, *Simulation* 71(2), 96-106

(8) Park C.S., Noh Y.D. (1987): A Port Simulation Model for Bulk Cargo Operations, *Simulation* 48(6), 236-246

(9) Ramani K.V. (1996): An Interactive Simulation Model for the Logistics Planning of Container Operations in Seaports, *Simulation* 66(5), 292-300

(10) Tahar M, Hussain K (2000): Simulation and analysis for the Kelang container terminal operation. *Logistics Information Management* 13(1), 14-20

(11) Yun W.Y., Choi Y.S. (1999): A Simulation Model for Container-Terminal Operation Analysis Using an Object-Oriented Approach, *International Journal of Production Economics* 59, 221-230

(12) Yuri M., Juri T., Eberhard B., Leonid N., Egils G., Elena V., Galina M., Jurijs P. (1998): A Modeling and Simulation Methodology for Managing the Riga Harbour Container Terminal, *Simulation* 71(2), 84-95