Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

CONTROLS: EMULATION TO IMPROVE THE PERFORMANCE OF CONTAINER TERMINALS

Csaba A. Boer
Yvo Saanen

TBA BV
Karrepad 2
2623 AP, Delft
THE NETHERLANDS

ABSTRACT

Nowadays container terminals are struggling with a continuously increasing volume. Therefore, they are searching for solutions to increase throughput capacity without expanding their physical footprint. Furthermore, they aim to increase their productivity on vessels in order to be able to handle bigger ships with larger call sizes in the same time frame. A terminal operating system (TOS) plays a major role in today's terminal operations as it supports planning, scheduling and equipment control. Recently more and more tasks are performed by the TOS – stowage planning, grounding decisions, equipment dispatching – and therefore, they need to be well-tuned to the operation, which remains a terminal specific characteristic. In this paper, an approach is presented to test and tweak the TOS and train operators on a virtual terminal. The implementation of this approach has been successfully applied during several TOS update or replacement projects for Rotterdam, Hong Kong, Norfolk, Virginia and Antwerp.

1 INTRODUCTION

A terminal operating system (TOS) is a software application supporting the planning, scheduling and equipment control activities of a container terminal and by this being responsible for accurate operations within the terminal (Agerschou et al. 2004). Recently different types of TOS exist on the market. The most popular ones are SPARCS (from NAVIS), SPACE/TRAFFIC from COSMOS, and CATOS from TSB.

On the one hand, the TOS is becoming increasingly importance for delivering high performance to the customer. On the other hand, the TOS needs to operate efficiently. Therefore, it is important to have a high quality TOS. These requirements motivated TBA BV to develop an emulation tool CONTROLS (which stands for CONtainer TeRminal Optimised Logistics Simulation) to allow for off-line experimentation with the TOS. Herewith a clear overview can be created in which parameters can be fine-tuned during software development, commissioning, and during training and operations. The tool itself together with the user of the tool form the success factors for the application.

The motivation behind working out this concept and developing and applying the container terminal emulation tool CONTROLS is:

- Allowing for realistic, comprehensive, safe and inexpensive testing of the TOS for both greenfield sites and for existing sites, implementing a new TOS, or receiving an upgrade
- Enabling off-line optimization of TOS parameters, by allowing forecasting simulation, replay animation, and long term projection simulation
- Allowing for realistic, real-time training, in a completely operational setting, for individual operational staff or complete teams

CONTROLS can support various types of operations and various TOS's. Accordingly, the tool consists of various modules that can be applied and configured for each terminal. For specific processes in terminals, the tool will be configured, and if necessary, additional customized components will be added.

This paper concerns the concept behind the emulation tool CONTROLS. The paper is structured as follows. First of all the concept of emulation in general is presented and emulation of container terminals is focused. Then the high level framework of CONTROLS is presented. CONTROLS is supported by several components such as Key Performance Indicator (KPI), SCENEMA, Graphical Editor, Experiment Monitor and Replay Tool. These components are presented in a nutshell. Next the features provided by CONTROLS are described. Finally some findings from several projects that have been carried out within CONTROLS are discussed and a conclusion is drawn.

978-1-4244-2708-6/08/$25.00 ©2008 IEEE

2 EMULATION OF A CONTAINER TERMINAL

Emulation refers to the ability of a software application or physical device to imitate another software application or device. Emulation is also considered as a certain phase within testing process of a controlled system.

The development of a complex systems (e.g. container terminal), which is controlled by a separate control system (e.g. TOS), may include one or more of the following phases, which aim to test the system during different design stages (Auinger, Vorderwinkler, and Buchtela 1999).

1. Full simulation: includes the simulation of both the complex system and the control system;
2. Real-time control: uses real complex system and simulates the control system;
3. Emulation: simulates the complex system and uses real control system;
4. Prototyping: involves tests with real complex system and real control system.

While full prototyping seems the most realistic testing possibility, it is quite expensive to build and experiment with the whole prototype system, especially because it involves the risk of errors if the possibilities of its design are not tested thoroughly beforehand. Full simulation involves lower costs, however, it may disregard some phenomena that are present in the real system, or it may contain additional factors that might influence the outcomes. Emulation and real-time control have the advantage in that they can be carried out in a cheaper way than full prototyping, they stay closer to reality, and they are, therefore, less time-consuming than full simulation (Verbraeck, Valentin, and Saanen 2000; Mueller 2001; Boer, Verbraeck, and Veeke 2002).

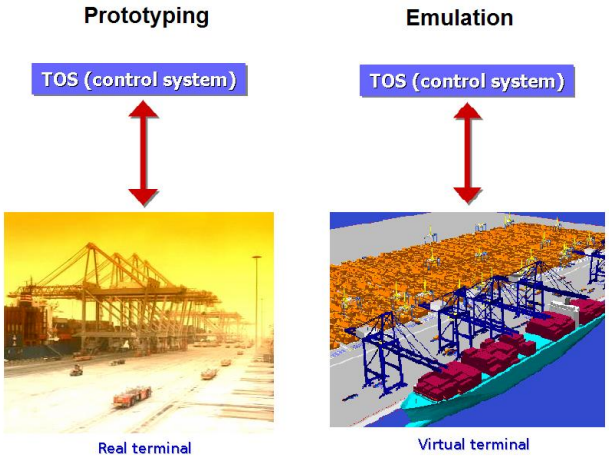

Figure 1: Emulation vs. Prototyping

Emulation of a container terminal (virtual representation of the container terminal and real TOS) acts as a real terminal, i.e. provides a valid representation of the physical processes at a terminal (equipment behaviour, driver behaviour, operational scenarios – gate arrivals, train arrivals, vessel arrivals). This can be linked to the TOS in such a way that the TOS treats the model as the "real world" (via existing interface between TOS and equipment), and it can be used to run operational scenarios, either as they occurred in the past or as configured by the user.

By examining the performance of the terminal controlled by TOS under various emulated conditions an assessment can be made of the system (TOS but also the operational processes) and its actual configuration. This assessment will be made by actually running an operation as would normally take place at a terminal as well, however, without moving containers in a physical way. Now, the TOS communicates with virtual drivers, clerks and other peripheral systems, rather than real ones. The communication protocol is completely identical to the communication in real-life. The emulation model includes a representation of all relevant processes at the terminal, e.g. the lay-out (yard, rail terminal and quay cranes), a model of the equipment (kinematics, driver behaviour, routing, disturbances, and availability), and performance measurement functionalities.

All performance relevant interactions that take place between the equipment at the terminal (including the gate and rail terminal) have to be defined and supported by the interface between the TOS and the emulation tool.

3 CONTROLS FRAMEWORK

CONTROLS is a software application which aims to create a simulated virtual container terminal. The framework of the application is well structured in multiple levels depicted as a pyramid in Figure 2.

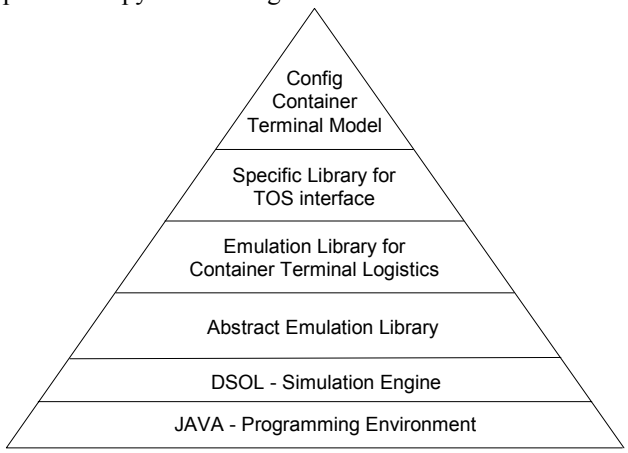

Figure 2: CONTROLS Model Pyramid

The lowest level in this pyramid is the programming environment that was chosen as a basis for CONTROLS.

Our choice was Java programming language, being nowadays one of the most commonly used programming languages (Eckel 2006).

For the simulation engine we have been using the D-SOL (Distributed Simulation Object Library). D-SOL is an open source, java-based, multi-formalism simulation suite that offers simulation services for the development of a fully distributed simulation model. This means that D-SOL allows (Jacobs, Lang, and Verbraeck 2002):

- an accurate specification of simulation model
- to separate the core simulation services from the model components that use it
- to separate of simulation and animation (even on separate computers)
- possibility to link to different information systems (databases, spreadsheets)
- different multi-formalism, such as discrete or continuous

On the top of D-SOL layer is the Abstract Emulation Library. This library contains abstract implementations of components that are necessary for emulation study. In this layer there is nothing about container terminal related implementation, which basically means that it might be reused for any emulation study (e.g. manufacturing). In this layer we introduce the concept of abstract vehicles, which later can be any specific equipment (e.g. car, truck, train, quay crane, rubber tired gantry, etc.). We tried to model it in a realistic way and even on the abstract level we made a distinction between the vehicle and its driver. The vehicle is represented as an object that can be moved only if it has a driver or controller (automated equipments). For equipment that is manually controlled, we introduced the concept of behaviour. Accordingly, a vehicle can be driven or operated in different ways, depending on the driver's or operator's behaviour. For this reason, we followed an agent-based representation of the drivers, which can have different behaviours depending on the circumstances. Furthermore, this layer includes the abstract implementation needed to connect different external control systems and abstract implementation for the animation.

Emulation Library for Container Terminal Logistics is one of the largest libraries we use containing the detailed implementation of all container terminal equipments. The implementation is based on the Abstract Emulation Library. In this library specific vehicles are implemented, such as quay crane (QC), rubber tired gantry (RTG), rail mounted gantry (RMG), straddle carrier (SC), reach stacker (RS), forklift (FL), terminal truck (TT) external truck (RT) and vessel (VL). For each type of vehicle we can apply different driving and operational behaviour. Moreover, this library includes the implementation of other container terminal specific objects, such as container, stack, interchange zones, etc.

The Specific Library for TOS interface contains the implementation of all interfaces that are needed to interoperate with the TOS. The TOS and the simulated equipment communicate via an interface that is already provided by the TOS supplier. The interface is identical to the interface used in the real operation, so that the TOS does not need to be adjusted, which is a core requirement to the emulation approach. This requirement is very important; otherwise changes may cause deviations from what would happen in a real operation. For each type of TOS we have a TOS interface library, e.g. SPARCS interface library or SPACE/TRAFFIC interface library.

At the top of the pyramid is the configuration of the virtual terminal requested by the customer. This is a thin layer on which the modeller configures the terminal model and the TOS interfaces according to the requirement. In CONTROLS everything is configurable: the road layout, the stack, buildings within the terminal, the number of used equipments, kinematics of equipments, TOS related information, etc. All this information is mainly represented as XML files and provided by the CONTROLS supported components. This layer uses these input configuration files to create the virtual terminal.

In the next section we present the CONTROLS-supported components that assist CONTROLS users in creating the input files, controlling the experiment, and analyzing the output files.

4 CONTROLS SUPPORTED COMPONENTS

CONTROLS is supported by several components, which can be used as standalone applications. Figure 3 depicts the relation between CONTROLS and supported applications.

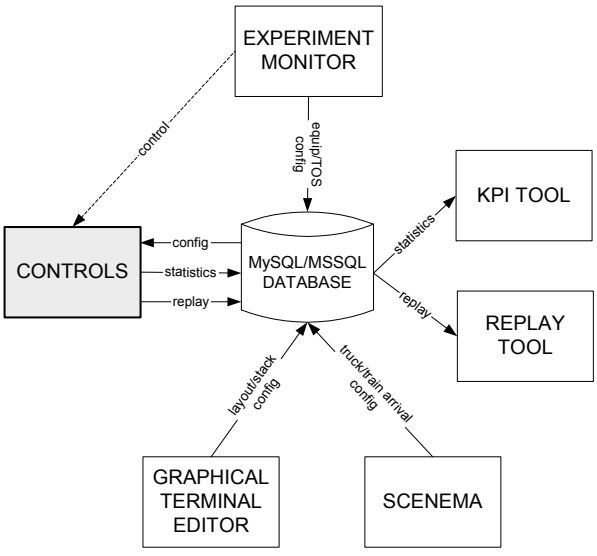

Figure 3: CONTROLS Supported Components

4.1 MySQL/MSSQL Database

CONTROLS can use different type of databases. We considered both the MySQL and the MS SQL Server. Both of these databases can be used to fulfil two important roles.

Firstly, it is used to store the input needed to configure a model. These configuration input data are provided by the Graphical Terminal Editor, SCENEMA and Experiment Monitor components.

Secondly, it is used to store output data needed to create statistics and replay data to rerun the whole experiment. While the statistical output data is used by KPI tool, the replay data is used by Replay tool.

4.2 KPI Tool

Typically, a TOS does not generate detailed statistics with regard to the equipment. Therefore, the CONTROLS model can be configured to publish enough information in the database that is then used by the Key Performance Indicator (KPI) tool to generate detailed statistics that go beyond equipments productivity (e.g. QC productivity, truck turn-around, RTG productivity, etc.). These statistics enable a more detailed analysis of the reasons for a certain productivity – for instance longer travel distances.

CONTROLS model publishes detailed order and status information for each operation, which are stored in the database. The KPI tool uses this information to create the desired statistical charts.

The KPI tool follows a so-called "shopping-cart" concept meaning that the user can select (choose and place statistical charts in the cart) and finally print (buy) the requested statistical charts. The statistical charts are printed in a structured form of a report in a format requested by the user (pdf, html, xls). The user has the possibility to create statistics for a single experiment or multiple experiments. The user can configure the already existing statistical objects, for instance one can create the statistical chart of average QC productivity when considering all cranes in the first 5 hours or the QC productivity for each crane during the whole experiment. The whole concept is based on selection and then configuration of the requested statistics. KPI tool allows both offline and online statistics.

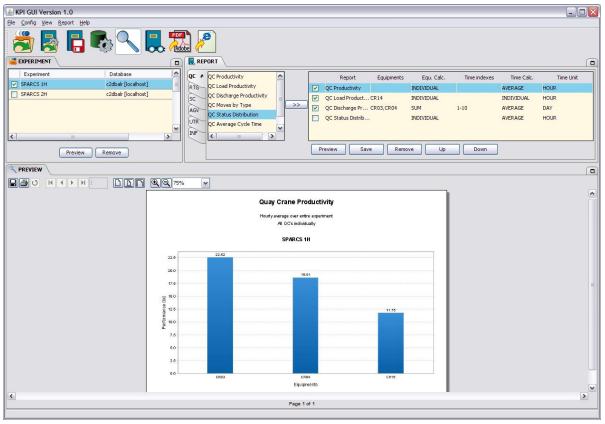

Figure 4: Snapshot KPI Tool

The following outputs are provided by the KPI tool (however this list is by no means exhaustive):

- QC (quay crane) productivity (load/discharge)
- QC status distribution
- QC cycle time
- SC (straddle carrier) productivity (productive and non-productive moves)
- Status of stacking equipments (idle, working, waiting for other equipments)
- Detailed equipment behaviour (average distance per move, average speed)
- Average filling rate of the yard
- Heat-maps showing the frequency of usage of certain area (e.g. route, stack or area under QC)

4.3 Experiment Monitor

The Experiment Monitor is a centralized controller application which is responsible to start and monitor emulation experiments with CONTROLS and TOS systems. The current version of monitor supports SPARCS TOS.

By using the Experiment Monitor the user has the possibility to configure the virtual container terminal model (e.g. number of equipment, equipments kinematics, etc.), select workstations on which CONTROLS and SPARCS instances are ready to be started, and afterwards start and control the experiment.

The data SPARCS uses in day-to-day operations can be found in the operational environment. The Experiment Monitor has an extra functionality to copy those data to a backup server, thus creating a so called 'snapshot'. On the backup server multiple snapshots can be stored. In order to perform an experiment, the monitor copies a selected single snapshot from the backup server to the emulation environment (both to SPARCS and CONTROLS workstations).

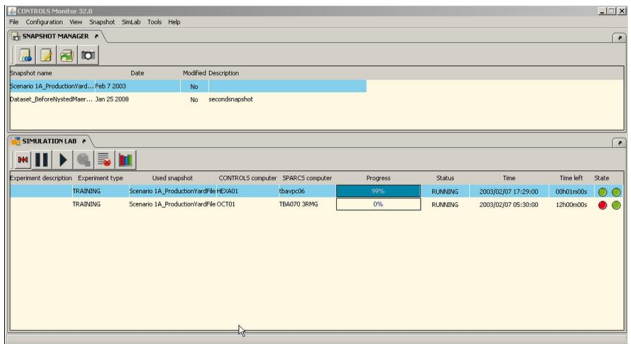

Figure 5: Snapshot Experiment Monitor

Next to monitor the status of the experiment the user has the possibility to obtain logging information from the CONTROLS model. Furthermore, one has the possibility to start the KPI Tool from the Experiment Monitor application in order to follow online statistics.

4.4 SCENEMA

In order to evaluate the parameter settings (e.g. allocation ranges), algorithms, and allocation of resources to points of work (vessel, rail, gate, yard), operational scenarios have to be defined. These scenarios are either real scenarios, meaning that it is a replay of a past existing operation, or they are representative created scenarios, for instance for peak circumstances (e.g. full yard, 20% volume growth, or a full berth after a storm) or breakdown situations.

Each scenario execution will consist of an initialization step during which the user (or the scenario manager) configures the initial yard (in principle loaded from the TOS), and plans the operation. These settings can then be stored, so that this experiment can be repeated without initializing again. After the initial yard is loaded, the experiment run will be executed. The experiment run will cover for instance 8 or 24 hours of operation – the ability to run longer experiments depends on the TOS since CONTROLS is capable of running as long as the TOS controls the operation. Attended runs may last as long as the user wants, as the emulation behaves as the real world would behave. Unattended runs, however, depend on the capability of the TOS to run without an operator controlling it.

In order to create scenarios for future operations, we have developed a scenario creation tool, called SCENEMA (Scenario Manager). The scenario manager can create all kind of realistic operational scenarios, based on container flow input. It can be used to create scenarios for future throughputs, or changed container flow parameters (more rail, more transhipment, etc.). The scenario manager converts the user input to consistent container flows, including vessel arrivals, gate arrivals, train arrivals, and container remaining in the yard for a certain dwell time. They are formatted such that the TOS can directly use them. These formats are generally EDI (Electronic Data Interchange), e.g. BAPLIE, MOVINS (seaside operation) and 417, 418 (train operation).

4.5 Graphical Terminal Editor

The Graphical Terminal Editor application is aimed at supporting the user in creating and modifying the layout of the yard (road infrastructure and stacks) in a visual, user-friendly manner.

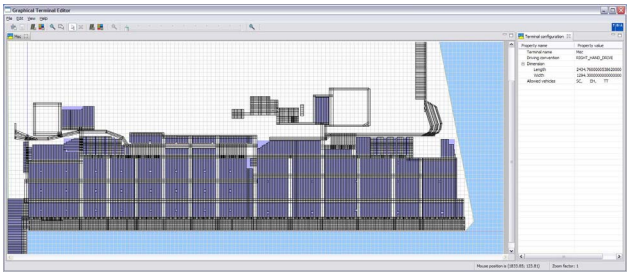

Figure 6: Snapshot Graphical Terminal Editor

In Graphical Terminal Editor the user creates the terminal by using container terminal objects, such as stack, road, quay, parking places, buildings, etc. The created configuration files (stack layout, road infrastructure layout) are stored in XML format. These configuration files are stored in a repository (e.g. database) and when creating an experiment, the CONTROLS applies these configuration files.

4.6 Replay Tool

We have distinguished two kinds of animation for simulation: *concurrent* and *post-processed* (Henriksen 1999). While in concurrent animation the animation is being displayed simultaneously with the simulation, in post-processed animation, state changes during the simulation run are saved to a repository (in our case database) and are used to drive the graphics after the simulation is over. Packages that provide a post-processed animation in the form of a 'replay tool' aim to eliminate the overhead created by the concurrent animation, which slows down the execution of the simulation model.

During the planning of the operations on the terminal (this typically occurs a day or shift in advance), CONTROLS will allow comparing different strategies that can be applied to increase the cost-efficiency of a particular upcoming operation. One has to think of the amount of equipment to be deployed per crane per period of time, the work sequence of work queues, the timeline of quay cranes and so on. The planner will have a number of CONTROLS instances available to run the alternative scenarios through, and compare the results. Accordingly, a dedicated powerful computer is used to run several

CONTROLS instances without animation and the result is monitored (by means of animation) on separate computers (see Figure 7).

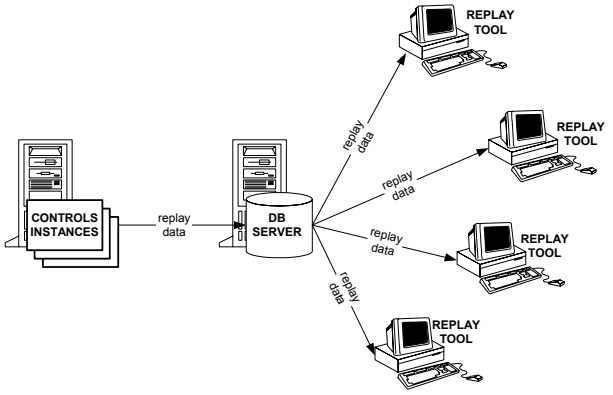

Figure 7: Post-processed Animation via Replay Tool

The post-processed animation provides an excellent approach to animate the operation on the terminal on different computers than the one on which the simulation is running (CONTROLS server). In this way several operators can follow from different angles the operations that concerns only their work.

This mechanism is implemented in Replay Tool, which next to concurrent animation allows the user to replay already recorded scenarios.

5 FEATURES PROVIDED BY CONTROLS

CONTROLS comprises all relevant components at the virtual terminal. It includes all container terminal equipments such as quay cranes, yard handling cranes (RTG or RMG), transportation equipments (straddle and shuttle carriers, terminal and road trucks), empty handlers (reach stacker, fork lift). Furthermore, it contains all relevant locations in the terminals, such as the gate and the truck transfer zones, the yard plus special stacks (e.g. reefer stack, empty stack, and place for OOG containers).

All equipment actually has a physical location and the physical boundaries are taken into consideration. The dynamic movements of equipment are based on equipment kinematics and on operation behaviour. Usually there are interactions between equipments at the terminal, for instance two SCs that want to access the same row in the stack. These interactions are either solved on the equipment level or by an intermediate semaphore. In most of the cases, as in reality, it will lead to one SC waiting. All equipment and its behaviour are designed and implemented in detail to keep their operation as close to reality as possible.

The configuration of the yard (road infrastructure, stack) is achieved by using the Graphical Terminal Editor. The configuration of the equipments is done by a configuration component which can be called either via CONTROLS GUI (when running one instance) or Experiment Monitor (when running more instances). Here the user can select the type of equipment one desires to use and the kinematics attributes for each equipment.

If concurrent animation is requested, CONTROLS shows a 3D animation panel within CONTROLS GUI. The user can disable the concurrent animation and enable the possibility to publish replay data for post-processed animation. In this case the data is published in a database and the operators can follow the animation from a remote computer (see Figure 7). The importance of this kind of animation is that the user does not bother the computer on which the simulation is running. Instead one uses another computer. Furthermore, the already-collected data can be always replayed for further investigation.

For both concurrent and post-processed animation, CONTROLS affords a pleasant 3D animation. The user can open multiple views (like placing multiple cameras to the terminal) and can save these views.

During the run the user has the possibility to examine certain statistics, such as the productivity of QCs or driven distances by certain TTs. The KPI tool provides statistics not only at the end of the run but also during the run. The online statistics (during the run) can help the user to take certain action if some equipments perform under expectations.

The emulation is capable of running up to 30 times faster than real-time depending on the size and complexity of the terminal. Although we can achieve a relatively high speed, it cannot be always used faster than real time because of the TOS. In order to run faster than real time there is a need for a time synchronization between CONTROLS and TOS. Although there are different mechanisms for time synchronization (Boer 2005), not all TOS systems provide this functionality. We achieved time synchronization with SPARCS. However, running faster than say two to three times real time might be dangerous because the TOS can not continue experimenting due to the heavy calculation needed for planning and scheduling.

CONTROLS can be used to validate old runs. Usually the TOS generates a large amount of log files that can be used to replay the whole operation. From these log files, however, we cannot predict certain important statistics, such as crane productivity, driven distance or status distribution of certain equipments. Therefore, we use these log files to replay the whole operation without or with minimal TOS support. We can compare the obtained results with the measurements done by the terminal (real statistics) and validate the CONTROLS model. If the measurements are similar (about 95%) then we consider the model valid and it can be used for the emulation study.

CONTROLS, being a virtual terminal, can provide additional functionality which in reality is not necessarily

present for the operators. This includes, for example, immediately finding and selecting a container, selecting (and following) a piece of equipment, selecting equipment and representing its main properties at that point of time (presenting the position of the equipment will be updated based on given time interval –e.g. every 1 second), showing all details information during the run for a certain group of equipments (e.g. QCs) and so on.

6 EXPERIENCES USING CONTROLS

In emulation projects using CONTROLS, the emulation has been mostly used to support the introduction of new TOS software or to update an existing one. In these projects emulation proved to be a very valuable tool, enabling the development team to solve about 95% of the errors that typically are found *after* going live.

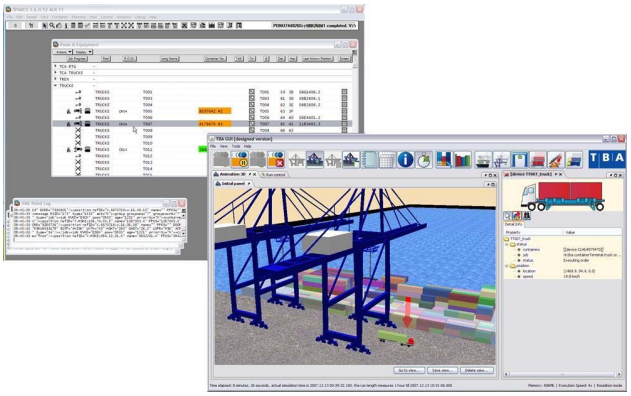

Figure 8: Running SPARCS and CONTROLS

Furthermore, the animation accompanying the emulation provided useful insight during the process, including for non-software experts. Where the customers of the system usually have to wait until the system goes live, they were now able to operate a virtual operation long before the system went into the real operation. This allowed future operators to be trained and also provided early feedback from the future users.

Finally, it enabled the team to stay focused on the performance of the terminal, or system. Where typically, the focus shifts from performance to "getting the things to work", a continued focus could be kept on performance (think of quay crane productivity, equipment utilization, stacking efficiency, truck service, etc.) because the tool environment allowed for this.

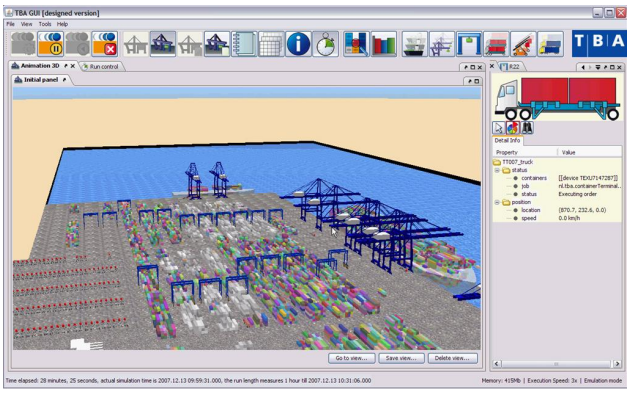

Figure 9: RTG Terminal in CONTROLS

In two of the cases we applied the emulation to improve productivity in a live environment, replaying past operations, evaluating different parameter settings. In one case, we aimed at improving truck turn around time by improving straddle carrier dispatching. It appeared from the experiments that the truck turn-around time during the peak – by just changing the dispatching algorithm and the decision factors – could be reduced by more than 30% on average, and the peaks by even 50%. At a minimal investment – it implied some changes to the TOS – the service to trucks could be improved drastically. In other words, the number of straddle carriers required to handle a certain truck peak could be reduced, saving labour and equipment costs.

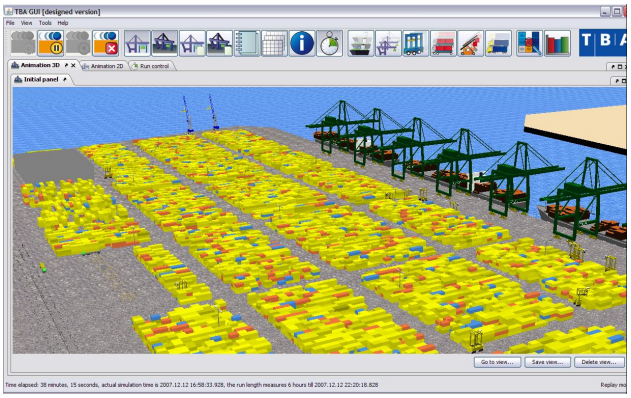

Figure 10: SC Terminal in CONTROLS

The second case aiming at improving productivity, showed that changes to grounding parameters – allocation filters for instance – in combination with equipment control parameters – for instance the weight factor of travel distance – can lead to significant improvements of the quay crane productivity (5-10% increase).

7 CONCLUSION

Due to the continuous volume increase, the need of handling more and more containers within given time window, and a limited yard place; larger vessels, new generation equipments, bigger terminals and more optimized operations are considered (Saanen 2004). Since a container terminal's performance relies on its TOS, it is vital to continuously improve the capability of the TOS. As a TOS gets more advanced and as terminals get more reliant on their TOS, the importance of sophisticated test, tuning and tweaking tools gets more vivid. Simply allowing TOS vendors to deliver new releases without running realistic scenarios, installing TOS without detailed performance testing during the commissioning, and developing TOS without the use of tools that make the performance metrics explicit, is something that can entail a huge risk and expenses.

In order to reduce the risk and expenses, an emulation tool has been developed, called CONTROLS. Being an emulation environment, it allows the user to experiment with a virtual environment and real TOS. Problems caused by TOS can be immediately recognized and solved in the virtual environment while staying close to reality in a cheap and safe manner.

Next to experimenting with an already existing terminal, representing a virtual terminal, it allows experimentation with the future terminal. Accordingly, CONTROLS also provides terminals with an opportunity to increase throughput without adding equipment or yard space: simply by doing *more with less*. As volumes increase, costs rise, and space is scarce, CONTROLS provides a very promising way out.

ACKNOWLEDGMENTS

We would like to thank our colleagues for their support in developing and testing CONTROLS and its supporting components.

REFERENCES

Agerschou, H., I. Dand, T. Sorensen, ,T. Ernst. 2004. *Planning and design of ports and maritime terminals*. 2nd ed. London: Thomas Telford Ltd.

Auinger, F., M. Vorderwinkler, and G. Buchtela. 1999. Interface Driven Domain-Independent Modelling Architecture for "soft-commissioning" and "reality in the loop". In *Proceeding of the 1999 Winter Simulation Conference*, ed. P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 798-805. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Boer, C. A., A. Verbraeck, and H. P. M. Veeke. 2002. The Possible Role of a Backbone Architecture in Real-Time Control and Emulation. In *Proceeding of the 2002 Winter Simulation Conference*, ed. E. Yücesan, C. H. Chen, J. L. Snowdon and J. M. Charnes, 1675-1682. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Boer, C. A. 2005. *Distributed Simulation in Industry*. ERIM Ph.D. Series Research in Management: Rotterdam, The Netherlands. Available via <http://hdl.handle.net/1765/6925> [accessed June 29, 2008].

Eckel, B. 2006. *Thinking in Java.* 4th ed. New York: Prentice Hall PTR.

Henriksen, J., O. 1999. General-Purpose Concurrent and Post-processed Animation with PROOF™. In *Proceeding of the 1999 Winter Simulation Conference*, ed. P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 176-181. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Jacobs, P. H. M., N. A. Lang, and A. Verbraeck. 2002. DSOL: A Distributed Java Based Discrete Event Simulation Architecture. In *Proceeding of the 2002 Winter Simulation Conference*, ed. E. Yücesan, C. H. Chen, J. L. Snowdon and J. M. Charnes, 793-800. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Mueller, G. 2001. Using emulation to reduce commissioning costs on a high speed bottling line. In *Proceeding of the 2001 Winter Simulation Conference*, ed. B.A. Peters, J.S. Smith, D.J. Medeiros, and M.W. Rohrer, Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Saanen, Y. 2004. *An Approach for Designing Robotized Marine Container Terminals.* Ph.D. thesis, Delft University of Technology, Delft, The Netherlands.

Verbraeck, A., E. Valentin, and Y.A. Saanen, 2000. Simulation as a Real-time Logistic Control System: AGV Control with Simple++. *The New Simulation in Production and Logistics – Prospects, Views and Attitudes*. pp. 245-255, Berlin, Germany.

AUTHOR BIOGRAPHIES

CSABA A. BOER is a senior product manager within TBA BV, one of the leading logistics and simulation consultancy firms in Europe. He is responsible for several products within the organization, one of which is CONTROLS. He holds a Ph.D. in Computer Science and Logistics from Erasmus University Rotterdam. His research interests include distributed simulation, distributed virtual environments, port logistics, and port simulation and emulation. His e-mail address is <csaba.boer@tba.nl>.

YVO SAANEN (MSc in Systems Engineering, PhD on the design and simulation of robotized container terminals, both Delft University of Technology) is managing director and founder (1996) of TBA, a leading simulation consultancy company in The Netherlands. He heads the department that supports ports and terminal operators all over the world in their design process of container terminals by means of simulation. In addition, Yvo Saanen is a lecturer at Erasmus University in Maritime Economics and Logistics. His e-mail address is <yvo.saanen@tba.nl>.