3

Discrete-Event Dynamic Systems Modelling Distributed Multi-Agent Control of Intermodal Container Terminals

Guido Maione
Technical University of Bari
Department of Environmental Engineering and Sustainable Development (DEESD)
Viale del Turismo, 8 – I74100 – Taranto
Italy

1. Introduction

Maritime intermodal container terminals (CTs) are complex hub systems in which multiple transport modes receive and distribute freight to various destinations. They can be considered as interchange places in a transport system network where seaways, railways, and motorways intersect.

Freight is usually organized, transported, stacked, handled and delivered in standard units of a typical container, which is called TEU (Twenty Equivalent Unit), 20 feet long, 8 feet wide, and 8 feet high. TEUs easily fit to ships, trains and trucks that are built and work for this kind of cargo units. Usually, ships travelling on long routes (e.g. from Taiwan to a Mediterranean Sea port) are called vessel or 'mother' ships, and ships offering service on short distances in a local area (e.g. from Turkey to Mediterranean ports in other countries) are called feeder ships. Vessel and feeder ships differ in size and container capacity.

A maritime CT is usually managed to offer three main services: a railway/road 'export cycle', when containers arrive to the terminal by trains/trucks and depart on vessel ships; a railway/road 'import cycle', when containers arrive on vessel ships and depart by trains/trucks; a 'transshipment cycle', when containers arrive on vessel (or feeder) ships and depart on feeder (or vessel) ships. These activities cause different concurrent and competition processes for the available resources. The aim is to achieve efficiency in flows of TEUs and in information distribution to guarantee fast operations and low costs.

Many problems were investigated and many of them have connections and cannot be solved separately. The main focused problems are: berth allocation of arriving ships; loading and unloading of ships (crane assignment, stowage planning); transfer of containers from ships to storage area and *viceversa* (quayside operation); stacking operations (yardside operation); transfer to/from other transport modes (landside operation); workforce scheduling. In a word, managing a CT requires:

- planning, monitoring and controlling all internal procedures for handling TEUs;
- optimizing handling equipment, human operators, and information and communication technologies (control software and hardware, PDAs, wireless sensors

Open Access Database www.i-techonline.com

Source: Robotics, Automation and Control, Book edited by: Pavla Pecherková, Miroslav Flídr and Jindřich Duník,
ISBN 978-953-7619-18-3, pp. 494, October 2008, I-Tech, Vienna, Austria

and actuators, etc.); optimization must take into account connections and relations between humans and hardware resources.

Then, an intelligent and reactive control system is required to meet terminal specifications, independently of disturbances and parameter variations. As a matter of fact, robustness is important to guarantee quick reaction to different phenomena perturbing normal (or steady-state) operating conditions. Perturbations may come from: increase of ship traffic volumes or urgent demands for service; infrastructure development (reduction/expansion of berthing or stacking spaces, changes in yard organization, acquisition of new resources, etc.); changes in routing of transport vehicles or traffic congestion inside the CT; faults and malfunctions or sudden lack of hardware resources.

Complexity of the issues involved justifies trends towards distributing the computing resources and decision controllers while aiming at robustness, and hence motivates heterarchical control by means of a *Multi-Agent System* (MAS). In a MAS, indeed, information, decision, and control are physically and logically distributed across agents. This chapter reports some preliminary results and ideas on how to model MAS controlling intermodal CTs, by using the DEVS-Scheme (Zeigler, 2000). The accurate model could be used to develop a simulation platform, which can be useful to test different control strategies to be applied in real cases. For example, the CT in Taranto (Italy) can be considered as a real system to be simulated in different operating scenarios.

The organization of this chapter is as follows. Section 2 discusses about literature contributions to modelling, simulation and control of CTs. Motivations are explained to use the methodology of *Discrete-Event Dynamic Systems* to represent the MAS architecture for simulation and control of complex CTs. Section 3 describes the typical services and processes to be guaranteed in an intermodal container terminal. Then, the basic components of the proposed MAS are presented, and their roles and relations are generally specified. Section 4 specifies how agents are modelled as atomic discrete-event dynamic systems and focuses on the interactions between agents, when containers are downloaded from ships to the terminal yard area. Section 5 gives some ideas and details about the plans of a simulation platform to test efficiency and robustness of the proposed MAS architecture. Section 6 overviews the benefits of the approach and enlightens the open issues.

2. Literature overview: motivation of DEVS modelling for MAS simulation and control

Planning processes and scheduling resources in a maritime CT pose very complex modelling and control problems to the scientific community. In particular, flexible and powerful modelling and simulation tools are necessary to represent intermodal hub systems made up of many different infrastructures and services. Modelling, simulation and control of CTs is a relatively new field of research. Several literature contributions developed models and simulation tools, but no standard exists to be applied to the different real terminals and scenarios. In the absence of standard tools, several research studies are based on discrete-event simulation techniques (Vis & de Koster, 2003), on mathematical models or empirical studies (Crainic et al., 1993; Peterkofsky & Daganzo, 1990; Gambardella et al., 1998).

Analytical models, based on different approaches, have been proposed as tool for the simulation of terminals useful to define the optimal design and layout, organization, management policies and control. A thorough literature review on modelling approaches is

given in (Steenken et al., 2004). As regards control issues, determining the best management and control policies is a challenging problem (Mastrolilli et al., 1998). Literature highlights two main classes of approaches: microscopic and macroscopic modelling approaches (Cantarella et al., 2006).

Microscopic modelling approaches are generally based on discrete-event system simulation that may include Petri Nets (Degano & Di Febbraro, 2001; Fischer & Kemper, 2000; Liu & Ioannou, 2002), object-oriented approaches (Bielli et al., 2006; Yun & Choi, 1999), and queuing networks theory approaches (Legato & Mazza, 2001). Even if it requires high computational effort, microscopic simulation allows the explicit modelling of each activity within the terminal as well as of the whole system by considering the single containers as entities. Then, it is possible to estimate performance as consequence of different system design and/or management scenarios.

Macroscopic modelling is suitable for terminal system analysis. It assumes continuous containers flow along the whole sequence of activities. This representation is useful and appropriate for supporting strategic decisions, system design, terminal layout, handling equipment investments. The most frequent problems in this class are berth planning, marshalling strategies, space allocation and system layout, handling equipment capacity and technologies. A network-based approach is presented in (Kozan, 2000) for optimising the terminal efficiency by using a linear programming method. In (de Luca et al., 2005) a macroscopic model is based on a space-time domain.

In the context of intelligent control of transport systems, this research contribution proposes a rigorous approach based on the *Discrete EVent System* (DEVS) specification technique (Zeigler et al., 2000) to completely and unambiguously characterize a MAS for controlling an intermodal container terminal. Namely, no standard exists for modelling and simulating complex MAS controlling CTs, but a generic and unambiguous framework is needed to describe the discrete and asynchronous actions of agents, and to guarantee modular analysis and a feasible computational burden. These requirements are easily enforced by the DEVS formalism. Connecting agents, each modelled as an atomic DEVS, makes the whole MAS represented as a DEVS, as the formalism is closed under coupling. Moreover, DEVS theory provides a strong mathematical foundation and models can be easily translated in a simulation environment.

The DEVS approach is then useful to develop a simulation platform to test the MAS efficiency in controlling the CT activities. In particular, both static and dynamically adapted decision strategies can be tested for the agents defining the MAS. Then, performance is measured in terms of commonly used indices (ship service time, throughput or lateness of containers, resource utilization, etc.) and other indices of the MAS efficiency (number of requests in negotiation, waiting time before decision, etc.). Performance can be evaluated both in steady-state operating conditions and in perturbed conditions, when disturbances or parameter variations occur.

Autonomous agents play as atomic DEVS dynamic systems. They exchange messages one with another to negotiate services in a common environment. Agents are local controllers, represented by software modules devoted to a hardware component or a specific function (like mediation between different agents). They behave autonomously and concurrently, by communicating each other to negotiate tasks, so exchanging messages and information to 'buy' or 'sell' services. Usually, there is no hierarchy between agents. In this context, cooperation may be explicitly designed or implicitly achieved through adaptation of the agents' decision mechanism, by using feedback of their effects. The general recognized

benefits of a MAS architecture are reducing the programming complexity, guaranteeing scalability and re-configurability, obtaining an intelligent and reactive control software.
The DEVS technique is fully compatible with the heterarchical design principles, it leads to MAS where all information and control functions are distributed across agents, and it is also suitable for a rigorous theoretical analysis of structural properties of the MAS. Moreover, the DEVS formalism is an interesting alternative to other recently proposed tools for MAS specification, e.g. the Unified Modeling Language (Huhns & Stephens, 2001) and Petri Nets (Lin & Norrie, 2001). This formalism is suitable to develop useful models both for discrete-event simulation and for implementation of the software controllers for the considered system.
As in MAS for manufacturing control (Heragu et al., 2002; Shen & Norrie, 1999), agents may use decision algorithms emulating micro-economic environments. Each 'buyer' agent uses a fictitious currency to buy services from other 'seller' agents which, on their turn, use pricing strategies. Sellers and buyers have to reach an equilibrium between conflicting objectives, i.e. to maximize profit and to minimize costs, respectively. Recently developed analytical models of negotiation processes (Hsieh, 2004), underline the need of a systematical analysis and validation method for distributed networks of autonomous control entities. Other researches have focused on the experimental and detailed validation of MAS on distributed simulation platforms (Schattenberg & Uhrmacher, 2001; Logan & Theodoropoulos, 2001).
This work proposes a DEVS model of a MAS architecture for controlling an intermodal container terminal system. In particular, a detailed DEVS model of the interactions, occurring between the agents concurrently operating during the critical downloading process of containers from a ship, is developed.
The author aims at giving a contribution to define a complete DEVS model to develop a detailed simulation platform for testing and comparing the proposed MAS with other centralized or distributed control architectures for intermodal container terminals. The simulation model could be used to design and test alternative system layouts and different control policies, to be used in standard or perturbed operating conditions.

3. Multi-agent system framework for intermodal container terminals

To define the autonomous agents operating and interacting in an intermodal container terminal environment, the main processes executed in the terminal area have to be examined to represent the most significant and critical aspects. These processes are associated to the offered services, syntethically described as follows.

3.1 Import, export and transshipment cycles in an intermodal container terminal

As recalled before, an intermodal terminal usually offers three different kinds of services interconnecting different transport modes:

1. an *import* cycle, when containers arrive on a vessel or feeder ship and depart by trains or trucks, corresponding to a transition from sea to railway or road modes;
2. an *export* cycle, when containers arrive by trains or trucks and depart on a vessel ship, for a transition from railway or road to sea mode;
3. a *transshipment* cycle, when containers arrive and depart by ship: cargo is moved from vessel to feeder ships to reach close destinations, or, *viceversa*, from feeder to vessel ships to reach far ports.

The possible flows of containers are synthetically described in Fig. 1, where TRS, IMP and EXP represent transhipment, import and export cycles, respectively, and RA and RO symbolize a railway or road transport mode. Note that full containers can be imported, exported or transshipped. Obviously, imported TEUs are only downloaded from ships, exported TEUs are only loaded on ships, while transshipped TEUs occur both processes. Empty containers are downloaded from feeder ships or arrive on trains or trucks, then they are loaded on vessel ships. So, these TEUs are transhipped or exported.

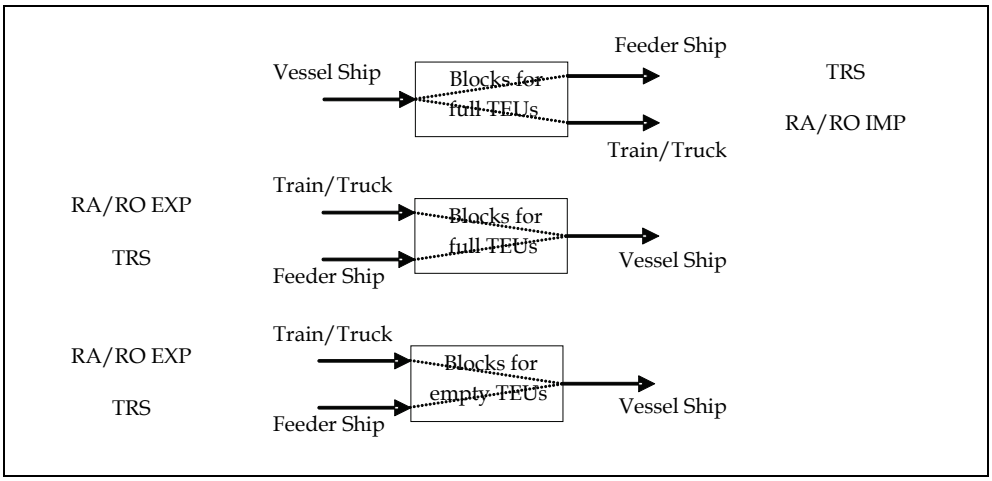

Fig. 1. Flows of containers in an intermodal container terminal

To execute these cycles, the companies managing container terminals provide activities like: loading/downloading containers on/from ships; delivering/picking containers to/from trucks or trains; stacking and keeping containers in dedicated areas, called blocks, in which the terminal yard is organized; transferring containers from ship/train/truck to yard blocks and backwards; inspecting containers for customs, safety and other requirements; consolidating, i.e. redistributing containers between blocks to allow fast retrieval.

Several dedicated or shared resources are used to execute the above processes: cranes (quay cranes, yard cranes, railway cranes, jolly mobile cranes); internal transport vehicles (trailers, automatically guided vehicles or AGVs); reach stackers for handling containers between trailers and trucks, and for dangerous or inspected containers; side loaders for handling empty containers; other special areas and infrastructures like quays and berths, lanes for internal transport, terminal gates, railway tracks; skilled human operators.

For example, the layout of the *Taranto Container Terminal* (TCT, see official website at *http://www.tct-it.com/*), located in the Taranto city harbour area in the south-east of Italy and currently managed by a private company, is organized as sketched in Fig. 2. A quay receives ships, yard stacking blocks are used for keeping full or empty TEUs. A gate (G) let trucks enter or exit the terminal, a railway connection allows trains to enter or exit. Special blocks are for parking of trailers (PARK), fuel station (FU), customs (EX), inspections, and other functions. A control centre (CC) is used to follow the processes. Stacking blocks are divided between blocks for full containers (from 11 to 46), blocks for empty containers (M's) and blocks for dangerous containers (DG). High-level planning operators schedule and monitor

all activities, while low-level quay and yard specialized operators execute the planned activities. Most of activities in TCT are for transshipment services.

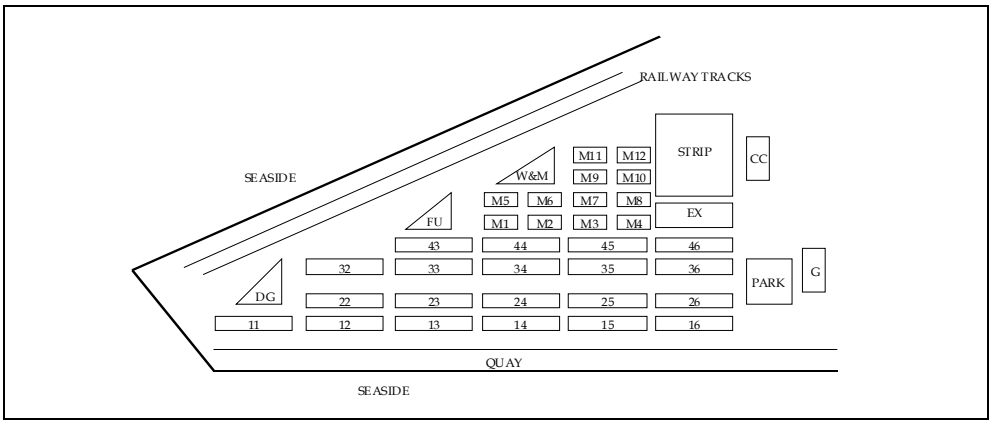

Fig. 2. Taranto Container Terminal organization

A road import cycle may be divided into three steps. Firstly, quay cranes download containers from a berthed ship to trailers, which transfer cargo into blocks. Here, yard cranes pick-up containers from trailers and stack them in assigned positions. Secondly, containers stay in the blocks for a certain time, while waiting for their destination; they are eventually relocated by yard cranes in a more proper position, according to a consolidation procedure which may use trailers to move containers between blocks. Thirdly and after consolidation, containers are loaded from blocks to trucks, which exit from the terminal gate. Similarly, in a railway import cycle, yard cranes pick-up containers from blocks and load them on trailers moving from yard to the railway connection, where special railway cranes pick-up containers to put them on departing trains.

In a road/railway export cycle, the sequence goes in the opposite direction: from the terminal gate or railway connection to blocks and then to vessel ships. If arriving on trucks, containers are transferred to yard blocks where they are picked up by yard cranes; then, they are stacked and consolidated; finally, they are moved by trailers to the quayside where quay cranes load them on ships. If containers arrive on trains, railway cranes load them on trailers for the transfer to yard blocks.

In a transshipment cycle, when a vessel ship arrives, containers are downloaded from ships to trailers using quay cranes, transferred to blocks using trailers, picked-up and stacked by yard cranes. After a consolidation and/or a delay in their position, containers are picked-up by trailers and transferred back to the quay area, where they are loaded on a feeder ship. The opposite occurs if a feeder arrives after a short trip and a vessel departs for a long one.

Hereinafter, the modelling focus will be on the first step of the import and transshipment cycles, when a downloading process from a ship is executed. Namely, this is considered as a critical phase for the terminal efficiency. But similar models and observations can be obtained for the subsequent steps and also for the export cycles.

3.2 Classification of agents in the MAS

Now, a MAS is specified modelling the negotiations occurring in an intermodal container terminal which is mainly devoted to transshipment cycles. The main considered agents are:

- the *Container Agent* (CA): each CA is an autonomous entity in charge of controlling the flow of a single container unit or a group of containers;
- the *Quay crane Agent* (QA): it is an autonomous controller for a (set of) quay crane(s) with the same performance characteristics, or with the same physical possibility to reach and serve ship bays;
- the *Trailer Agent* (TA): it is an entity associated to a (set of) trailer(s) with the same performances, or with the same reachable portions of quay or yard spaces;
- the *Yard crane Agent* (YA): it manages the control of a (set of) yard crane(s) guaranteeing the same performances, or associated to the same yard blocks;
- the *Railway crane Agent* (RA): it is a software controller of a (set of) railway crane(s), used to receive containers from trailers and to load them on trains, or to deliver containers from trains to trailers;
- the *Truck Agent* (KA): it follows the operations executed by a (set of) truck(s), entering or leaving the terminal by the gate.

If related to a set of containers, a CA represents units physically stowed in a ship bay when considering the downloading process or units stacked in yard blocks when considering the loading process.

In import processes, the CA should identify the most suitable quay crane to download containers from ship, then the most suitable trailers to transport containers to their assigned yard blocks, and finally the most proper yard cranes to pick-up and stack containers in their assigned block positions. All these choices result from negotiations between the CA and several QAs, TAs, and YAs. The CA has also responsibilities in export processes: it selects the yard cranes to pick-up containers from blocks, trailers to transport them to the quay area, and quay cranes to load them into their assigned bay-row-tier location in the ship. In this case, the CA negotiates with YAs, TAs, and finally YAs.

During consolidation, containers are sometimes moved around and relocated to new positions in yard blocks, such that subsequent export operations are optimized or made easier. These moves are the consequence of other negotiations of CAs with YAs and TAs.

The decisions taken by a CA are based upon real-time updated information received from agents of the alternative available cranes and trailers.

The global control of the activities in the terminal emerges from the behaviour of concurrently operating agents. The dynamical interaction between agents has to be analysed to specify the desired global system behaviour. For instance, the precedent observations lead us to examine interactions between a CA and several QAs, TAs, and YAs for downloading, transferring, and stacking containers, or for picking, transferring, and loading containers. Interactions exist also between a CA and YAs, TAs, and RAs/KAs when railway/road transport modes are considered.

The interaction between agents is usually based on a negotiation mechanism, which is typically organized in the following steps. *Announcement*: an agent starts a bid and requires availability to other agents for a service. *Offer*: the agent requests data to the agents which declared availability. These data regard the offered service the queried agents can guarantee. *Reward*: the agent selects the best offer between the collected replies from the queried agents, and sends a rewarding output message. *Confirmation*: the agent waits for a confirmation message from the rewarded agent, after which it acquires the negotiated service. If confirmation is delayed or does not arrive, then the agent selects another offer in the rank or starts the bid over again.

In this work, the focus is on the interactions between a CA with QAs, TAs and YAs for downloading containers from ships to yard blocks during transhipment cycles. The reason is twofold: transhipment is the main and more complex service in intermodal container terminals (e.g., see the TCT case), and the downloading part has critical effects on terminal efficiency. The developments may be easily extended to other negotiations between agents for the loading part of transshipment cycle, export and import cycles or other processes.
The agents' negotiations and decisions are only limited by:

- constraints of terminal spaces and resources (e.g., the limited number of quay cranes that may physically serve a fixed ship bay, considering that quay cranes are sequentially lined up and move along fixed tracks; the limited number of yard cranes serving blocks);
- fixed working schedules for downloading/loading processes (they often establish the number of containers moved for each ship bay, their exact location in the ship hold or cover, the sequence of handling moves, sometimes even the preferred handling quay crane).

Agents' decisions are fulfilled by human operators devoted to the associated resources (i.e. crane operators, trailer/truck drivers). The network of interacting agents may appear and behave as a unique distributed supervisor for the physical terminal system.

4. Discrete-event systems modelling of agents' dynamics

Now, each agent in the previously identified classes is described as an atomic DEVS (Zeigler, 2000). Namely, all agents interact by transmitting outputs and receiving inputs, which are all considered as event messages. Events are instantaneous, then timed activities are defined by a start-event and a stop-event.
For each agent, internal events are triggered by internal mechanisms, external input events (i.e. inputs) are determined by exogenous entities, for example other agents, and external output events (i.e. outputs) are generated and directed to other entities.
These external or internal events change the agent state. Namely, an agent stays in a state until either it receives an input X or the time scheduled by a time advance function *ta* elapses (this time specifies the time before the occurrence of an internal event I). In the first case, an external transition function δ_{ext} determines the state next to the occurrence of the received input; in the second case, an internal transition function δ_{int} gives the state next to the occurrence of the internal event. An output function λ is used to generate the reactions of the agent, each output being indicated with a symbol Y.
It is important to note that the DEVS formalism makes a difference between the *total state*, **q**, and the *sequential state*, **s**. This latter refers to the transition mechanism due to internal events. It is based on the current value of the so-called *status*, i.e. the condition in which an agent stays between two consecutive events during negotiations, and other characteristic information *inf* peculiar to the considered agent:

$$\mathbf{s} = (status, inf) \tag{1}$$

The total state is composed by **s**, the time *e* elapsed since the last transition, and some additional information like the decision logic **DL** currently used by the agent to rank and choose the offers received by other agents during negotiation:

$$\mathbf{q} = (\mathbf{s}, e, \mathbf{DL}) \tag{2}$$

For a CA, **s** may include information on: the current container position; the quay cranes available for negotiating downloading/loading operations; the trailers available for negotiating the transport from quay to yard or *viceversa*; the yard cranes available for negotiating pick-up and stacking operations (from trailer to a block position or backwards); the time scheduled in current state before the next internal event, if no external input event occurs.
For QAs, YAs, TAs, RAs, KAs, the sequential state may include the time prospected before the next internal event and the queued requests coming from CAs for availability, for data about offered service, for the confirmation of assigned service, etc..
To summarize, each agent can be represented as an atomic DEVS in the following way:

$$\boldsymbol{A} = < \mathbf{X}, \mathbf{Y}, \mathbf{S}, \delta_{int}, \delta_{ext}, \lambda, ta > \tag{3}$$

where **X** is the set of input events, **Y** is the set of output events, **S** is the set of sequential states, δ_{int}: $\mathbf{S} \rightarrow \mathbf{S}$ is the internal transition function, δ_{ext}: $\mathbf{Q}\times\mathbf{X} \rightarrow \mathbf{S}$ is the external transition function, $\mathbf{Q} = \{\mathbf{q} = (\mathbf{s}, e, \mathbf{DL}) \mid \mathbf{s}\in\mathbf{S}, 0 \leq e \leq ta(\mathbf{s})\}$ is the set of total states, λ: $\mathbf{S} \rightarrow \mathbf{Y}$ is the output function, ta: $\mathbf{S} \rightarrow \Re_0^+$ is the time advance function, with $\Re_0^+$ set of positive real numbers with 0 included.
Here, the *status*-transitions triggered by events are examined since the *status* is considered as the main component of the sequential state. This assumption let the dynamics of each agent be described by *status*-transitions.
The same DEVS methodology can be applied to represent interactions between agents in the loading part of a transhipment cycle, when containers are moved from yard blocks to ships, or in import or export cycles.

4.1 Dynamics of interactions between agents in a downloading process

The MAS can be regarded as 'container-driven' because each container follows a sequence of handling operations executed by terminal resources. The container is the main entity flowing in the system and flows of containers are controlled by agents.
To download containers from a ship bay, transport and stack them into a yard block, each CA follows three phases in negotiation. Firstly, it interacts with QAs to choose the quay crane for downloading. Secondly, with TAs to select the trailers for moving the containers from the quay to the yard area. Thirdly, with YAs to determine the yard cranes for stacking.
Then, it is assumed that the CA firstly communicates exclusively with QAs, then with TAs only, finally with YAs. The negotiation is based on data about the offered services: a QA gives the estimated time to wait before the associated quay crane can start downloading containers, and the estimated time to execute the operation; a TA the estimated time to wait a trailer, and the estimated time for the transport task; a YA the estimated time for the yard crane to be ready close to the block, and the estimated time for the stacking task.
Note that, for the loading part of the transshipment cycle from yard to ship, the CA will sequentially interact with YAs, TAs, and QAs.

4.1.1 Interactions between a CA and QAs

For $t<t_{C0}$ let the CA, say *C*, associated with a generic container be in a quiescent *status* (QUIESC) and let it begin its activity at t_{C0} (input event X_{C0}). Then *C* spends the time interval $[t_{C0}, t_{C1}]$ to send outputs $Y_{C01}, Y_{C02},\ldots, Y_{C0q}$ at instants $t_{01}>t_{C0}, t_{02},\ldots, t_{0q}=t_{C1}$. These messages

request the availability to all the q alternative QAs of quay cranes that can serve the container. The sequence of requests cannot be interrupted by any internal or external occurrence. For sake of simplicity, instead of modelling a sequence of q different and subsequent *status*-values, REQQAV represents the whole duration of the activity and it is assumed that *C* makes transition at t_{C1} (internal event I_{C1}).

In [t_{C1}, t_{C2}] agent *C* waits for answers (WAIQAV) from QAs. Namely, the request *C* transmits to each QA may queue up with similar ones sent by other CAs. Next transition occurs at t_{C2} when either *C* receives all the answers from the queried QAs (X_{C1}), or a specified time-out of WAIQAV expires before *C* receives all the answers. In case it receives no reply within the time-out (I_{C2}), *C* returns to REQQAV and repeats the request procedure. In case of time-out expiration and some replies received (I_{C3}), *C* considers only the received answers to proceed. The repeated lack of valid replies may occur for system congestion, for crane failures or communication faults, or for other unpredictable circumstances. In all cases permanent waits or deadlocks may occur. To avoid further congestion and improve system fault-tolerance, time-outs are used and *C* is allowed to repeat the cycle REQQAV-WAIQAV only a finite number of times, after which *C* is replaced by another software agent.

If all or some replies are received before the time-out expiration, *C* starts requesting service to the $g \leq q$ available QAs at t_{C2}. In [t_{C2}, t_{C3}] *C* requests information to these QAs by sending them outputs Y_{C11}, Y_{C12}, …, Y_{C1g} at instants $t_{11}>t_{C2}$, t_{12},…, $t_{1g}=t_{C3}$. As the sequence of requests cannot be interrupted, REQQSE *status* is referred for the whole activity: at t_{C3} agent *C* is assumed to make transition (I_{C4}).

Then, agent *C* spends [t_{C3}, t_{C4}] waiting for offers from the available QAs (WAIQOF), as the request *C* transmits to each QA may queue up with those sent by other CAs. Next transition occurs at t_{C4} when either *C* receives all the answers from the queried QAs (X_{C2}) or a time-out of WAIQOF expires. In case no reply is received within the time-out (I_{C5}), *C* returns to REQQSE and repeats the procedure. In case of time-out expiration and some replies are received (I_{C6}), *C* considers only the received offers to select the crane. Again, to avoid congestion, *C* repeats the cycle REQQSE-WAIQOF a finite number of times, then it is discharged.

Once received the offers from QAs, *C* utilizes [t_{C4}, t_{C5}] to take a decision for selecting the quay crane (TAKQDE). At t_{C5} the decision algorithm ends (I_{C7}), after selecting a QA and building a rank of all the offers from QAs.

Subsequently, *C* reserves the chosen crane by transmitting a booking message (Y_{C2}) to the corresponding QA. So *C* takes [t_{C5}, t_{C6}] for communicating the choice to the 'winner' QA (COMCHQ). At t_{C6} the communication ends (I_{C8}). Now, the selected QA has to send a rejection, if there is a conflict with another CA, or a booking confirmation (X_{C5}). Hence, *C* uses [t_{C6}, t_{C7}] to wait for a confirmation from the selected QA (WAIQCO). The confirmation is necessary because the availability of the cranes can be modified by actions of CAs other than *C* during the decision interval, and the selected crane can be no longer available. If *C* receives a rejection (X_{C3}), or does not receive any reply within a time-out (I_{C9}), it returns to COMCHQ, and sends a new request of confirmation to the second QA in the decision rank. If *C* has no other alternative destinations and the rejection (X_{C4}) or the time-out (I_{C10}) occurs, it returns to REQQAV and repeats the negotiation. Note that WAIQAV, WAIQOF and WAIQCO cannot lead to indefinite circular waits (deadlocks), thanks to the time-out mechanism.

At t_{C7}, after receiving a confirmation (X_{C5}) from the selected QA, *C* makes a transition to issue a downloading command (DWNLDG). It takes the interval [t_{C7}, t_{C8}] to issue the command Y_{C3} for the quay crane downloading the container.

4.1.2 Other interactions

When, at time t_{C8}, the downloading command is complete (I_{C11}), *C* starts the second negotiation-phase with TAs for a trailer to carry the container to its assigned block.

This new negotiation follows the same procedure as in the interaction with QAs. Agent *C* requests and waits for availability, requests information and waits for offers about trailer service, evaluates and ranks the offers to take a decision, communicates the choice to the best offering TA, and waits for a confirmation/rejection. This can be noted from Fig. 3, where the second part of the *status*-transition graph has the same repeated structure as in the first part, and the *status*-values have the same meaning. Also, time-outs are used to limit waiting and the repeated requests. This fact guarantees the model is modular, which is very important for simulation and control purposes.

When a confirmation is received, *C* makes a transition to issue a transport command (TRANSP), and the container is loaded on the vehicle associated to the selected TA.

Then, *C* starts the third negotiation-phase with YAs for a yard crane to stack the container in an assigned position inside a specified block. Again, due to the modularity of the approach, the sequence of allowed *status*-values follows the same protocol, and, finally, if a confirmation is received, *C* makes a transition to issue a stacking command (STCKNG). When the command is complete, *C* gets back to QUIESC.

From t_{C24} to the beginning of the next negotiation cycle (if any) for downloading, consolidating or loading another container, *C* stops making decisions, receiving and sending messages, and remains quiescent. The associated container is downloaded, transported and stacked in a block where it waits for the next destination (a new block or a ship bay): all these processes do not involve agent activities. Only when they are over, *C* is ready to start a new negotiation for the same container. If faults occur to the selected cranes or trailer, *C* remains in QUIESC and there is no need to restart negotiations with QAs, TAs, and YAs. Terminal operators manage the repair process and when the normal operating conditions are restored, the container can be handled by the selected resources.

Fig. 3 depicts the complex interaction dynamics previously described for the negotiation between a CA with QAs, TAs, and YAs. Circles represent the CA *status*-values, and the period spent in each *status* is indicated aside of it. Arrows represent the (internal or input) events, labelling the arrows themselves (the event time is indicated below the event symbol). The outputs, directly associated with *status*-values, are encapsulated in the circles. The output function simply defines outputs for the allowed *status*-values. The time advance function gives the residual time $ta(\mathbf{s})$ in state **s** before the scheduled occurrence of next internal invent. For instance, at the time t^* of entering a waiting *status*, the related time-out fixes the maximum time to wait T_w, such that $ta(\mathbf{s}) = t^*+T_w-t$, where t is the current time.

To synthesize, one can use a 'macro-status' for each of the negotiation phases and depict the diagram of Fig. 4. Macro-status *QA-neg* represents all the diagram corresponding to the first negotiation-phase between the CA and QAs. In the same way, *TA-neg* and *YA-neg* aggregate the parts of diagram in Fig. 3, which are related to the second and third negotiation-phases of the CA with TAs and YAs, respectively.

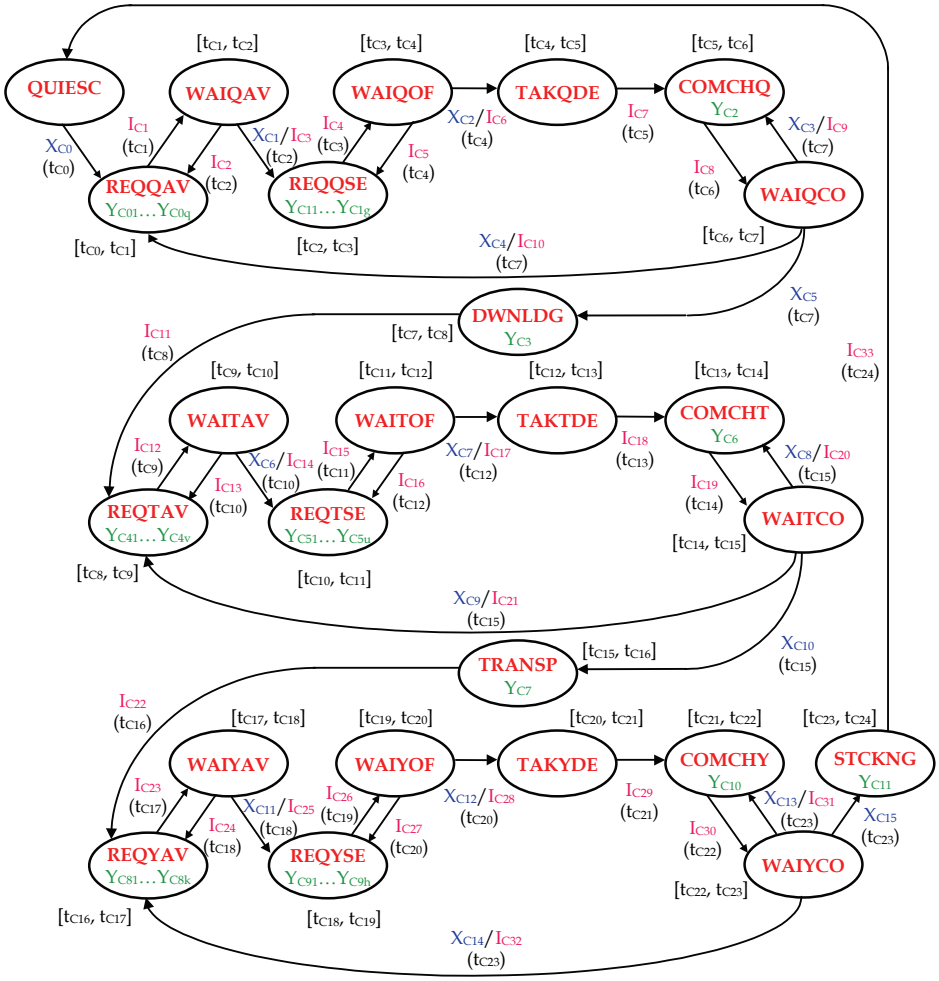

Fig. 3. Downloading in a transshipment cycle: negotiation of a CA with QAs, TAs, and Yas

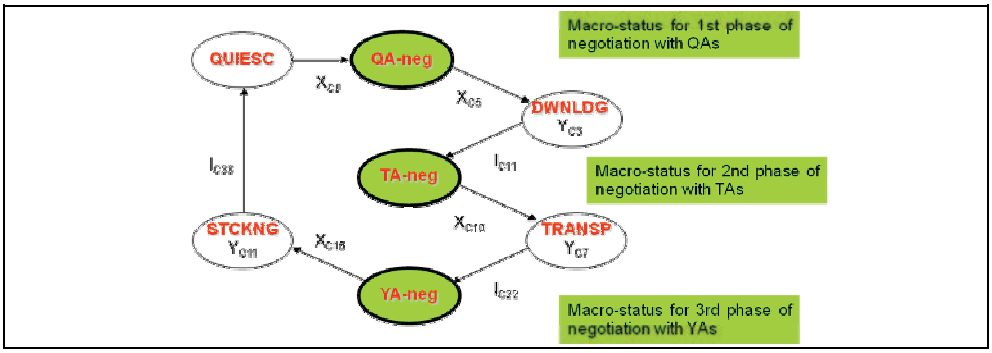

Fig. 4. Synthesized diagram of negotiations of a CA with QAs, TAs, and YAs

4.1.3 Remarks about negotiation

As Fig. 3 shows, the CA may receive confirmation from a QA, TA or YA after several successive loops COMCHQ-WAIQCO, COMCHT-WAITCO or COMCHY-WAIYCO.
Time-outs can bring the CA back to REQQAV from WAIQAV if no availability signal comes from QAs, or from WAIQCO if all selected QAs rejected the selection. The CA can also go back to REQTAV from WAITAV if no availability signal comes from TAs, or from WAITCO after rejection from all selected TAs. Finally, the CA comes back to REQYAV from WAIYAV if no availability signal comes from YAs, or from WAIYCO after rejection from all selected YAs.
Other time-outs rule the loops between WAIQOF and REQQSE, between WAITOF and REQTSE, and between WAIYOF and REQYSE.
One could merge the requests for availability and for information to get a more compact representation. But this would imply a lower effectiveness in reducing CA waiting times and in preventing deadlocks. Namely, the detailed model separates and reduces the effects of delays and losses of messages due to communication faults or to the unavailability of cranes and trailers, when particular conditions occur (faults, overloading conditions, etc.). Moreover, the number of *status*-loops and the consequent delays in decision are reduced if the described detailed model is adopted.

4.2 Specification of the DEVS model of a container agent

On the basis of the negotiation mechanism described in section 4.1 and by Fig. 3, the components of the DEVS model of a CA are identified. Table 1 reports and explains all the admissible *status*-values.
The sequential state **s** = (*C-status*, *C-inf*) of a container agent is defined as function of the current *status*-value, with *C-status* belonging to the set of all admissible values:

$$\begin{aligned} C\text{-}status \in \{&\text{QUIESC, REQQAV, WAIQAV, REQQSE, WAIQOF, TAKQDE, COMCHQ,} \\ &\text{WAIQCO, DWNLDG, REQTAV, WAITAV, REQTSE, WAITOF, TAKTDE, COMCHT,} \\ &\text{WAITCO, TRANSP, REQYAV, WAIYAV, REQYSE, WAIYOF, TAKYDE, COMCHY,} \\ &\text{WAIYCO, STCKNG}\} \end{aligned} \tag{4}$$

and as function of other information *C-inf* about the CA, which is defined as follows:

$$C\text{-}inf = (p, AQ, AT, AY, ta(\mathbf{s})) \tag{5}$$

which, in turn, depends on the current position p of the container (on ship, picked by quay crane, on trailer, picked by yard crane, in the yard block); the set AQ of alternative quay cranes available for the currently negotiated downloading operations, or the set AT of trailers available for the currently negotiated transport tasks, or the set AY of yard cranes available for the currently negotiated stacking tasks; the time $ta(\mathbf{s})$ scheduled in current state **s** before the next internal event occurrence.
A detailed description of inputs, outputs and internal events is reported in Tables 2, 3 and 4, respectively, for sake of clarity. It can be easily derived from the interactions between a CA and QAs, TAs, and YAs, as explained in section 4.1 and indicated in Fig. 3.
Most inputs received and outputs sent by a CA obviously correspond to outputs coming from and inputs to other agents, respectively. Then, these messages will concur to define the DEVS atomic models of the associated agents. More specifically, the agents interacting with

a CA will change their *status*, then their sequential state, on the basis of the inputs received by the CA. Moreover, they will generate outputs by using their specific output functions to answer the CA during the different negotiation-phases.

Phase of negotiation	*Status*	*Activity Description*
	QUIESC	Agent quiescent (inactive)
First phase	REQQAV	Request availability to all alternative QAs
	WAIQAV	Wait for availability from QAs
	REQQSE	Request service to available QAs
	WAIQOF	Wait for offers from available QAs
	TAKQDE	Rank QAs and take decision for the best QA
	COMCHQ	Communicate choice to selected QA
	WAIQCO	Wait confirmation/rejection from selected QA
	DWNLDG	Command selected QA to download container
Second phase	REQTAV	Request availability to all alternative TAs
	WAITAV	Wait for availability from TAs
	REQTSE	Request service to available TAs
	WAITOF	Wait for offers from available TAs
	TAKTDE	Rank TAs and take decision for the best TA
	COMCHT	Communicate choice to selected TA
	WAITCO	Wait confirmation/rejection from selected TA
	TRANSP	Command selected TA to transport container
Third phase	REQYAV	Request availability to all alternative YAs
	WAIYAV	Wait for availability from YAs
	REQYSE	Request service to available YAs
	WAIYOF	Wait for offers from available YAs
	TAKYDE	Rank YAs and take decision for the best YA
	COMCHY	Communicate choice to selected YA
	WAIYCO	Wait confirmation/rejection from selected YA
	STCKNG	Command selected YA to stack container

Table 1. *Status*-values for a Container Agent

Finally, note how the successive inputs, outputs and internal events of the CA in the three subsequent negotiation-phases repeat with the same role, which gives the DEVS model a structure that can be easily implemented in simulation.

DEVS models can be also specified for the other agents, by following the same methodology. In particular, the sequential state of each QA, TA or YA will include information about the queued requests coming from different CAs competing for the same controlled quay crane, trailer or yard crane.

Phase of negotiation	*Input*	*Time*	*Event Description*
	X_{C0}	t_{C0}	Start of negotiation activity for a new operation
First phase	X_{C1}	t_{C2}	Last of *q* replies for availability received from QAs
	X_{C2}	t_{C4}	Last of *g* replies for offer received from QAs
	X_{C3}	t_{C7}	Rejection & other QAs in the CA rank
	X_{C4}	t_{C7}	Rejection & no other QA in the CA rank
	X_{C5}	t_{C7}	Confirmation from selected QA
Second phase	X_{C6}	t_{C10}	Last of *v* replies for availability received from TAs
	X_{C7}	t_{C12}	Last of *u* replies for offer received from TAs
	X_{C8}	t_{C15}	Rejection & other TAs in the CA rank
	X_{C9}	t_{C15}	Rejection & no other TA in the CA rank
	X_{C10}	t_{C15}	Confirmation from selected TA
Third phase	X_{C11}	t_{C18}	Last of *k* replies for availability received from YAs
	X_{C12}	t_{C20}	Last of *h* replies for offer received from YAs
	X_{C13}	t_{C23}	Rejection & other YAs in the CA rank
	X_{C14}	t_{C23}	Rejection & no other YA in the CA rank
	X_{C15}	t_{C23}	Confirmation from selected YA

Table 2. Input events received by a Container Agent in negotiation with QAs, TAs, and Yas

Phase of negotiation	*Output*	*Time*	*Event Description*
First phase	$Y_{C01}, Y_{C02}, \ldots, Y_{C0q}$	$t_{01}>t_{C0}, t_{02}, \ldots, t_{0q}=t_{C1}$	Requests of availability to *q* QAs
	$Y_{C11}, Y_{C12}, \ldots, Y_{C1g}$	$t_{11}>t_{C2}, t_{12}, \ldots, t_{1g}=t_{C3}$	Requests of information to *g* available QAs
	Y_{C2}	t_{C6}	Choice communication to the selected QA
	Y_{C3}	t_{C8}	Command for downloading container
Second phase	$Y_{C41}, Y_{C42}, \ldots, Y_{C4v}$	$t_{41}>t_{C8}, t_{42}, \ldots, t_{4v}=t_{C9}$	Request of availability to *v* TAs
	$Y_{C51}, Y_{C52}, \ldots, Y_{C5u}$	$t_{51}>t_{C10}, t_{52}, \ldots, t_{5u}=t_{C11}$	Requests of information to *u* available TAs
	Y_{C6}	t_{C14}	Choice communication to the selected TA
	Y_{C7}	t_{C16}	Command for transporting container
Third phase	$Y_{C81}, Y_{C82}, \ldots, Y_{C8k}$	$t_{81}>t_{C16}, t_{82}, \ldots, t_{8k}=t_{C17}$	Request of availability to *k* YAs
	$Y_{C91}, Y_{C92}, \ldots, Y_{C9h}$	$t_{91}>t_{C18}, t_{92}, \ldots, t_{9h}=t_{C19}$	Requests of information to *h* available YAs
	Y_{C10}	t_{C22}	Choice communication to the selected YA
	Y_{C11}	t_{C24}	Command for stacking container

Table 3. Output events sent by a Container Agent in negotiation with QAs, TAs, and Yas

Phase of negotiation	*Internal Event*	*Time*	*Event Description*
First phase	I_{C1}	t_{C1}	End of request for availability of QAs
	I_{C2}	t_{C2}	Time-out & no availability signal received from QAs
	I_{C3}	t_{C2}	Time-out & *g* availability signals received from QAs
	I_{C4}	t_{C3}	End of request for offered service from available QAs
	I_{C5}	t_{C4}	Time-out & no offer received from the *g* available QAs
	I_{C6}	t_{C4}	Time-out & $oq \leq g$ offers received from available QAs
	I_{C7}	t_{C5}	End of decision for choosing quay crane (agent)
	I_{C8}	t_{C6}	End of choice communication to the selected QA
	I_{C9}	t_{C7}	Time-out & no confirmation received from the selected QA & ranked offers are available from other QAs
	I_{C10}	t_{C7}	Time-out & no confirmation received from the selected QA & no ranked offers are available from other QAs
Transition to 2nd phase	I_{C11}	t_{C8}	End of downloading command
Second phase	I_{C12}	t_{C9}	End of request for availability of TAs
	I_{C13}	t_{C10}	Time-out & no availability signal received from TAs
	I_{C14}	t_{C10}	Time-out & *u* availability signals received from TAs
	I_{C15}	t_{C11}	End of request for offered service from available TAs
	I_{C16}	t_{C12}	Time-out & no offer received from the *u* available TAs
	I_{C17}	t_{C12}	Time-out & $ot \leq u$ offers received from available TAs
	I_{C18}	t_{C13}	End of decision for choosing trailer (agent)
	I_{C19}	t_{C14}	End of choice communication to the selected TA
	I_{C20}	t_{C15}	Time-out & no confirmation received from the selected TA & ranked offers are available from other TAs
	I_{C21}	t_{C15}	Time-out & no confirmation received from the selected TA & no ranked offers are available from other TAs
Transition to 3rd phase	I_{C22}	t_{C16}	End of transport command
Third phase	I_{C23}	t_{C17}	End of request for availability of YAs
	I_{C24}	t_{C18}	Time-out & no availability signal received from YAs
	I_{C25}	t_{C18}	Time-out & *h* availability signals received from YAs
	I_{C26}	t_{C19}	End of request for offered service from available YAs
	I_{C27}	t_{C20}	Time-out & no offer received from the *h* available YAs
	I_{C28}	t_{C20}	Time-out & $oy \leq h$ offers received from available YAs
	I_{C29}	t_{C21}	End of decision for choosing yard crane (agent)
	I_{C30}	t_{C22}	End of choice communication to the selected YA
	I_{C31}	t_{C23}	Time-out & no confirmation received from the selected YA & ranked offers are available from other YAs
	I_{C32}	t_{C23}	Time-out & no confirmation received from the selected YA & no ranked offers are available from other YAs
Transition to QUIESC	I_{C33}	t_{C24}	End of stacking command

Table 4. Internal events in a Container Agent in negotiation with QAs, TAs, and YAs

Namely, let Q, T and Y be three of such agents. Then, each agent will be characterized by a queue $\mathbf{c}_1$ for the availability requests associated to one of the messages: Y_{C0i} ($1 \le i \le q$) arriving to Q, Y_{C4i} ($1 \le i \le v$) arriving to T, Y_{C8i} ($1 \le i \le k$) arriving to Y; a queue $\mathbf{c}_2$ for the information requests associated to one of the messages: Y_{C1i} ($1 \le i \le g$) arriving to Q, Y_{C5i} ($1 \le i \le u$) arriving to T, Y_{C9i} ($1 \le i \le h$) arriving to Y; a queue $\mathbf{c}_3$ for the confirmation requests associated to messages: Y_{C2} arriving to Q, Y_{C6} arriving to T, Y_{C10} arriving to Y. Moreover, the state of Q, T, or Y will be defined by the current number of containers served by the associated quay crane, trailer or yard crane.

5. Ideas for simulation and evaluation of efficiency and robustness of the MAS control

The DEVS atomic models can be integrated to obtain a complete networked system, which can be used as a platform for simulating the MAS architecture for controlling an intermodal container terminal. E.g. the TCT in Taranto can be used as a test-bed.
In this context, it is possible to simulate not only the dynamics of terminal activities, the flow of containers, and the utilization of terminal resources (cranes, trailers, human operators, etc.), but also the efficiency of the MAS and its agents (flow of event messages, *status* transitions, waiting loops, etc.).
Then, two types of performance indices can be defined. Namely, it is possible to measure the following conventional indices: the total number of handled (imported, exported, transshipped) containers; the average throughput, during downloading (from ship to yard) or loading (from yard to ship) processes; the average lateness of containers in the terminal; the utilization of resources; the ship turn-around time, i.e. the average time required to serve a ship for downloading and loading containers. Moreover, it is possible to measure the behaviour of the MAS and the efficiency of the agents' decision policies by means of: the average number of requests for each negotiation; the number of repeated negotiation loops of *status*-values before a final decision is taken by a CA, expressed in percentile terms with respect to the total number of operations executed by every CA. The lower is this index, the better is the agent capability to obtain a service at the first request. The higher is the value of the index, the higher is the lack of feasible replies due to congestion of the other agents or of the communication system.
More specifically, the terminal performance measures can be evaluated both in steady-state operating conditions and in perturbed conditions. Perturbations may arise from: hardware faults or malfunctions; abrupt increase/decrease of containers to be handled due to changes in maritime traffic volumes; sudden increase/reduction of yard space; traffic congestion of trailers; congestion, delays, message losses in the communications between agents.
For example, the private company managing TCT usually plans and controls the activities to serve one ship at a day. This is due to the fact that ship arrivals are known and scheduled in advance with ship agencies. But the company itself has recently foreseen a traffic increase in the following years, due to expected cargo movements coming from China and eastern countries to the Mediterranean Sea. Then, it is quite reasonable to think about working days in which more than one ship is berthed and served at the same time. In this case, at least two ships berthed to the quay would give a big perturbation to the required operations and terminal efficiency. Fig. 5 shows a snapshot of a discrete-event simulation made in this condition (2 ships in quay in the schematic view of the terminal), by using a conventional centralized control architecture, based on the current policies used in the terminal. The performance obtained were much lower than in standard conditions.

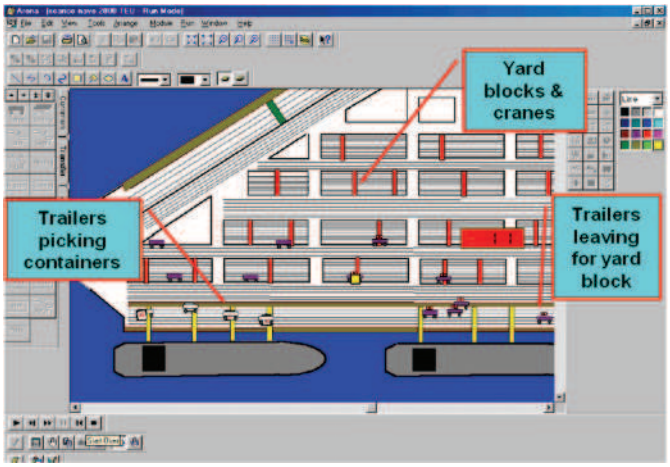

Fig. 5. Simulation snapshot of TCT in perturbed conditions – centralized control

Then, it is important to use distributed MAS control architectures and to measure robustness of agents' decision laws, to see how they dynamically react to disturbances and parameter variations, and eventually to adapt them. The adaptation aims to make the autonomous agents learn the most appropriate decision laws in all terminal conditions.

In this sense, the decision logic **DL** of a CA could be partly constant to encapsulate the most reliable strategies, and partly adaptable according to a learning algorithm. The constant part is constituted by a set of R heuristic decision rules, each related to a different evaluation parameter provided by QAs, TAs, and YAs. Some decision rules may be more effective in perturbed conditions (anomalies, faults, congestions), whereas a tradeoff between different rules may be more appropriate in other cases. Therefore, weights assigned to the rules represent the adjustable part of **DL** = $\{\alpha_1, \alpha_2, \ldots, \alpha_R\}$, where each α_j (j = 1, ..., R) specifies the factor weighting the role of the j-th heuristic in the global decision criterion. Then, an evolutionary algorithm can be used to adapt factors. In this way, in any operating condition, the worst performance of an agent should never be significantly lower than the performance of the worst decision rule.

To conclude, the discrete-event simulation platform also allows the comparison of alternative types of control architectures which can be defined by:

- a static MAS in which CAs use static logics based on heuristic decision parameters (estimated delivery time of the requested task, distance of cranes or trailers);
- a dynamic MAS in which CAs take decisions by fuzzy weighted combinations of heuristic criteria; the weights are adapted by an evolutionary algorithm;
- other distributed control architectures.

6. Conclusion

A MAS architecture was proposed for controlling operations in intermodal container terminal systems. The autonomous agents are represented as atomic DEVS components. The interactions between agents are modelled according to the DEVS formalism to represent negotiations for tasks when downloading containers from ships to yard stacking blocks. The

developed model can be easily extended to describe other processes, like loading containers from yard to ships, redistributing containers in the yard, import or export cycles.
An accurate DEVS model of the MAS can be used in a detailed simulation environment of a real system (the TCT in Taranto), which allows the measurement of standard terminal performance indices and of the efficiency of the MAS in real scenarios.
Moreover, open issues are testing and comparing static MAS and dynamically adapted MAS, if for example evolutionary adaptation mechanisms are used, especially with reference to different operating scenarios and to possible perturbations with respect to steady-state operating conditions.

7. References

Bielli, M.; Boulmakoul, A. & Rida, M. (2006). Object oriented model for container terminal distributed simulation. *European Journal of Operational Research*, Vol. 175, No. 3, pp. 1731-1751, ISSN 0377-2217

Cantarella, G. E.; Cartenì, A. & de Luca, S. (2006). A comparison of macroscopic and microscopic approaches for simulating container terminal operations, *Proceedings of the EWGT2006 International Joint Conferences*, pp. 556-558, ISBN 88-901798-2-1, Bari, Italy, 27-29 September 2006, 01Media, Molfetta (Bari), Italy

Crainic, T. G.; Gendreau, M. & Dejax, P. (1993). Dynamic and stochastic models for the allocation of empty containers. *Operations Research*, Vol. 41, No. 1, pp. 102-126, ISSN 0030-364X

de Luca, S.; Cantarella, G. E. & Cartenì, A. (2005). A macroscopic model of a container terminal based on diachronic networks, *Second Workshop on the Schedule-Based Approach in Dynamic Transit Modelling*, Ischia, Naples, Italy, 29-30 May 2005

Degano, C. & Di Febbraro, A. (2001). Modelling Automated Material Handling in Intermodal Terminals, *Proceedings 2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM'01)*, Vol. 2, pp. 1023-1028, ISBN 0-7803-6737-5, Como, Italy, 8-12 July 2001, IEEE, Piscataway, NJ, USA

Fischer, M. & Kemper, P. (2000). Modeling and Analysis of a Freight Terminal with Stochastic Petri Nets, *Proceedings of the 9th IFAC Int. Symp. Control in Transportation Systems*, Vol. 2, pp. 195-200, ISBN-13 978-0-08-043552-7, Braunschweig, Germany, 13-15 June 2000, Eds. Schnieder, E. & Becker, U., Elsevier-Pergamon, Oxford, UK

Gambardella, L. M.; Rizzoli, A. E. & Zaffalon, M. (1998). Simulation and planning of an intermodal container terminal. *Simulation, Special Issue on Harbour and Maritime Simulation*, Vol. 71, No. 2, pp. 107-116, ISSN 0037-5497

Heragu, S. S.; Graves, R. J.; Kim, B.-I. & Onge, A. St. (2002). Intelligent Agent Based Framework for Manufacturing Systems Control. *IEEE Trans. Systems, Man, and Cybernetics - Part A: Systems and Humans*, Vol. 32, No. 5, pp. 560-573, ISSN 1083-4427

Hsieh, F.-S. (2004). Model and control holonic manufacturing systems based on fusion of contract nets and Petri nets. *Automatica*, Vol. 40, No. 1, pp. 51-57, ISSN 0005-1098

Huhns, M. N. & Stephens, L. M. (2001). Automating supply chains. *IEEE Internet Computing*, Vol. 5, No. 4, pp. 90-93, ISSN 1089-7801

Kozan, E. (2000). Optimising container transfers at multimodal terminals. *Mathematical and Computer Modelling*, Vol. 31, No. 10-12, pp. 235-243, ISSN 0895-7177

Legato, P. & Mazza, R. M. (2001). Berth planning and resources optimisation at a container terminal via discrete event simulation. *European Journal of Operational Research*, Vol. 133, No. 3, pp. 537-547, ISSN 0377-2217

Lin, F. & Norrie, D. H. (2001). Schema-based conversation modeling for agent-oriented manufacturing systems. *Computers in Industry*, Vol. 46, No. 3, pp. 259-274, ISSN 0166-3615

Liu, C. I. & Ioannou, P. A. (2002). Petri Net Modeling and Analysis of Automated Container Terminal Using Automated Guided Vehicle Systems. *Transportation Research Record*, No. 1782, pp. 73-83, ISSN 0361-1981

Logan, B. & Theodoropoulos, G. (2001). The distributed simulation of multi-agent systems. *Proceedings of the IEEE*, Vol. 89, No. 2, pp. 174-185, ISSN 0018-9219

Mastrolilli, M.; Fornara, N.; Gambardella, L. M.; Rizzoli, A. E. & Zaffalon, M. (1998). Simulation for policy evaluation, planning and decision support in an intermodal container terminal, *Proceedings Int. Workshop "Modeling and Simulation within a Maritime Environment"*, pp. 33-38, ISBN 1-565-55132-X, Riga, Latvia, 6-8 September 1998, Eds. Merkuryev, Y., Bruzzone, A. & Novitsky, L., Society for Computer Simulation International, Ghent, Belgium

Peterkofsky, R. I. & Daganzo, C. F. (1990). A branch and bound solution method for the crane scheduling problem. *Transportation Research Part B: Methodological*, Vol. 24B, No. 3, pp. 159-172, ISSN 0191-2615

Schattenberg, B. & Uhrmacher, A. M. (2001). Planning Agents in James. *Proceedings of the IEEE*, Vol. 89, No. 2, pp. 158-173, ISSN 0018-9219

Shen, W. & Norrie, D. H. (1999). Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey. *Knowledge and Information Systems, An International Journal*, Vol. 1, No. 2, pp. 129-156, ISSN 0219-1377

Steenken, D.; Voss, S. & Stahlbock, R. (2004). Container terminal operation and operations research - a classification and literature review. *OR Spectrum*, Vol. 26, No. 1, pp. 3-49, ISSN 0171-6468

Vis, I. F. A. & de Koster, R. (2003). Transshipment of containers at a container terminal: An overview. *Europ. Jour. Operational Research*, Vol. 147, No. 1, pp. 1-16, ISSN 0377-2217

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*, Vol. 59, No. 1-3, pp. 221-230, ISSN 0925-5273

Zeigler, B. P.; Praehofer, H. & Kim, T. G. (2000). *Theory of Modelling and Simulation*, Academic Press, ISBN 0-12-778455-1, New York, 2nd edition

Robotics Automation and Control

Edited by Pavla Pecherkova, Miroslav Flidr and Jindrich Dunik

ISBN 978-953-7619-18-3
Hard cover, 494 pages
Publisher InTech
Published online 01, October, 2008
Published in print edition October, 2008

This book was conceived as a gathering place of new ideas from academia, industry, research and practice in the fields of robotics, automation and control. The aim of the book was to point out interactions among various fields of interests in spite of diversity and narrow specializations which prevail in the current research. The common denominator of all included chapters appears to be a synergy of various specializations. This synergy yields deeper understanding of the treated problems. Each new approach applied to a particular problem can enrich and inspire improvements of already established approaches to the problem.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Guido Maione (2008). Discrete-Event Dynamic Systems Modelling Distributed Multi-Agent Control of Intermodal Container Terminals, Robotics Automation and Control, Pavla Pecherkova, Miroslav Flidr and Jindrich Dunik (Ed.), ISBN: 978-953-7619-18-3, InTech, Available from: http://www.intechopen.com/books/robotics_automation_and_control/discrete-event_dynamic_systems_modelling_distributed_multi-agent_control_of_intermodal_container_ter

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2008 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.