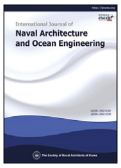
Available online at www.sciencedirect.com

ScienceDirect

International Journal of Naval Architecture and Ocean Engineering 8 (2016) 83−101
http://www.journals.elsevier.com/international-journal-of-naval-architecture-and-ocean-engineering/



Event-based scenario manager for multibody dynamics simulation of heavy load lifting operations in shipyards

Sol Ha [a], Namkug Ku [b,*], Myung-Il Roh [c]

[a] Department of Ocean Engineering, Mokpo National University, South Korea
[b] Department of Naval Architecture and Ocean Engineering, Dong-eui University, South Korea
[c] Department of Naval Architecture and Ocean Engineering & Research Institute of Marine Systems Engineering, Seoul National University, South Korea

Received 13 July 2014; revised 11 October 2015; accepted 21 October 2015
Available online 18 January 2016

Abstract

This paper suggests an event-based scenario manager capable of creating and editing a scenario for shipbuilding process simulation based on multibody dynamics. To configure various situation in shipyards and easily connect with multibody dynamics, the proposed method has two main concepts: an Actor and an Action List. The Actor represents the anatomic unit of action in the multibody dynamics and can be connected to a specific component of the dynamics kernel such as the body and joint. The user can make a scenario up by combining the actors. The Action List contains information for arranging and executing the actors. Since the shipbuilding process is a kind of event-based sequence, all simulation models were configured using Discrete EVent System Specification (DEVS) formalism. The proposed method was applied to simulations of various operations in shipyards such as lifting and erection of a block and heavy load lifting operation using multiple cranes.

Keywords: Scenario management; Discrete event simulation; DEVS (Discrete EVent System Specification); Multibody dynamics; Shipbuilding process

1. Introduction

Requests for accurate dynamic analysis using a simulation tool have been increasing in many engineering fields, including in the shipbuilding industry, as shown in Fig. 1. Unlike the conventional mechanical systems such as car and machinery, all ships and offshore structures are different from each other in purpose, shape and size. Thus, even though process planning may be set up based on past experience of similar ships and offshore structures, many problems which are not expected in advance may occur during production.

Moreover, according to the recent increase in the demand for offshore plants and new concept ships, reviews of new manufacturing methods to confirm their availability and safety have been performed frequently with their dynamic analysis in shipyards. Past studies on shipbuilding process and virtual manufacturing in shipyards were not focused on dynamic analysis of certain partial operations, but entire shipbuilding process planning and job assignment (Hwang et al., 2014; Lee et al., 2014). Since various existing simulation tools based on multibody dynamics focus on conventional mechanical systems (Cha et al., 2010a), such as machinery, cars, and spacecraft, there are some problems in the application of these simulation tools to shipbuilding domains due to the absence of specific items in naval architecture and ocean engineering, such as hydrostatic, hydrodynamic, wake, and mooring forces. Therefore, some recent studies focused on developing a simulation tool for the shipbuilding process based on the multibody dynamics theorem (Cha and Roh, 2010; Cha et al., 2010a, 2010b, 2012).

Since all ships and offshores differ in purpose, their production processes in shipyards also differ. Even if the mix of

* Corresponding author.
E-mail address: knk80@deu.ac.kr (N. Ku).
Peer review under responsibility of Society of Naval Architects of Korea.

http://dx.doi.org/10.1016/j.ijnaoe.2015.10.004

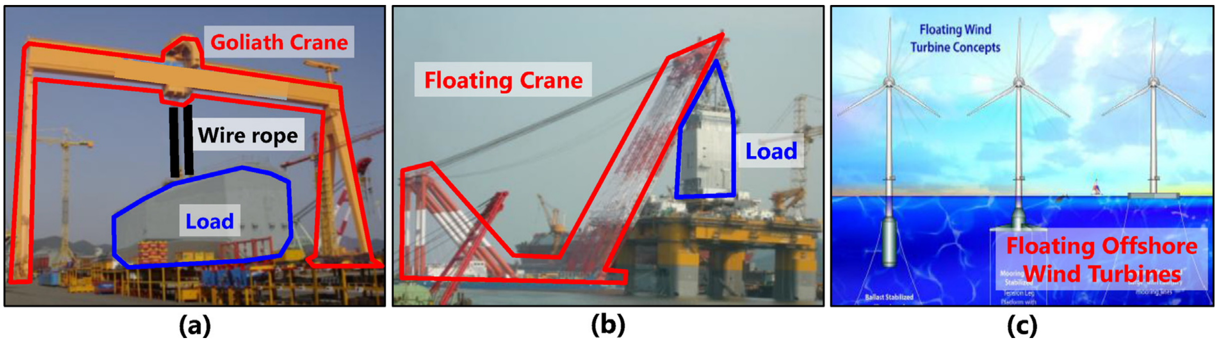

Goliath Crane
Wire rope
Load
(a)
Floating Crane
Load
(b)
Floating Wind Turbine Concepts
Floating Offshore Wind Turbines
(c)

Fig. 1. Various types of mechanical systems in the shipbuilding industry: (a) goliath crane, (b) floating crane, and (c) floating offshore wind turbines.

vessel on the order is familiar, the master schedule could be non-repeatable due to a different order in which the different ships appear in the order book. Thus, developing a procedure for describing the production of ships may be costly and time-consuming. In this paper, a scenario manager capable of creating and editing a scenario for the simulation of shipbuilding processes is proposed. To describe the process of shipbuilding in a multibody dynamics simulator, the user-defined inputs to the simulator were analyzed, and the shipbuilding scenario was broken down into modularized units. Due to the discontinuous process of shipbuilding, event-based formalism was applied to the scenario manager. It is expected that the scenario manager might be very useful even if there are not unusual ship-types in the orderbook.

The remainder of this paper is as follows. Section 2 reviews previous studies related to this paper. In Section 3, the developed multibody dynamics simulator for the shipbuilding process is briefly introduced. Section 4 describes the key ideas of the proposed scenario manager and its implementation, and its application to shipbuilding follows in Section 5. Finally, Section 6 summarizes this study and briefly discusses the next study.

2. Related works

2.1. Dynamics simulator and its application to shipbuilding process

There are various open-source-based or commercial software programs that are based on the multibody dynamics theorem. Since most of them are general-proposed programs and do not include all the force modules as hydrostatics, hydrodynamics, etc., there are a little cases of their application to shipbuilding production process simulation.

ADAMS (Automatic Dynamic Analysis of Mechanical Systems) is a software system that consists of a number of integrated programs that aid an engineer in performing three-dimensional kinematic and dynamic analysis of mechanical systems based on multibody system dynamics (Orlandea et al., 1977; Schienhlen, 1990). Various external forces can also be applied to multibody systems, but hydrostatic and hydrodynamic forces, which are the dominant forces exerted on the floating platform and frequently used in shipyards, cannot be handled by ADAMS. ODE (Open Dynamics Engine) is an open-source library for simulating multibody dynamics (Smith, 2006). Similar to ADAMS, ODE derives equations of motion for multibody systems using augmented formulation. However, ODE also cannot handle hydrostatic and hydrodynamic forces. RecurDyn (FunctionBay, 2003) is a three-dimensional simulation software program that combines dynamic response analysis and finite element analysis tools for multibody systems. It is two to 20 times faster than other dynamic solutions because of its advanced fully recursive formulation. RecurDyn cannot also handle hydrostatic and hydrodynamic forces.

Unlike these software that are based on multibody dynamics, MOSES (Multi-Operational Structural Engineering Simulator) is a simulation software that can analyze the movement of a single body in a fluid by applying hydrostatic force and hydrodynamic force to it (Ultramarine, 2013). With this software, a multibody system that is connected in a restrictive condition cannot be simulated because a connective relation between the bodies is not supported, but a simulation that considers hydrostatic force and hydrodynamic force from external forces is possible. Thus, MOSES is often used for ocean shipyard simulation for floating single bodies.

In other areas of studies not using these software, some researches related to shipbuilding domains have been conducted in the past. Cha et al. proposed and developed a simulation framework for dynamic analysis of shipbuilding production process (Cha and Roh, 2010; Cha et al., 2010a, 2010b, 2012). Related works described above are summarized and compared with this study in Table 1.

From the view of scenario management, ODE and MOSES have not tried to fully handle the scenario. It means that the user should configure and manage the simulation time and give certain action to change the motion of rigid body. In order to change the motion of rigid body, for example, the user should act an external force to certain rigid body manually. Moreover, these all simulators have not tried to handle discontinuous-state variables and event- and state-triggered conditions, so it is difficult to flexibly generate a scenario related to various application domains that requires the simulation in shipyards because the process of shipbuilding is a kind of event-based sequence. Thus, this study focused on a concept of managing a multibody dynamics simulation

Table 1
Related works.

Items	This study	Cha et al.	ADAMS	ODE	RecurDyn	MOSES
Multibody Formulation	Recursive Formulation	Embedding Technique	Augmented Formulation	Augmented Formulation	Recursive Formulation	General Newton−Euler Equation
Various Joints	O	O	O	O	O	X
Flexible Body	X	X	O	X	O	X
Hydrostatic Force	O	O	X	X	X	O
Linearized Hydrodynamic Force	**O**	**O**	X	X	X	**O**
Scenario Management	**O**	**O**	**O**	X	**O**	△

scenario based on a Discrete EVent System Specification (DEVS) formalism (Zeigler et al., 2000).

Cha et al. (2010d) already proposed the concept of an event-based scenario manager for the shipbuilding process. Cha and Roh (2010) simulated a block erection process that is frequently performed in shipyards using the simulation kernel developed by Bang. Cha et al. (2010) proposed basic simulation components to provide schemes and modules for the efficient development of application systems using the simulation kernel. They considered the necessity of a reusable simulation model in shipbuilding simulation, and proposed the "elementary simulation model," which is suitable for the development of a sequentially proceeding simulation, and the "simulation model coupling and scenario management scheme."

However, even though they considered the reuse of the simulation model, the proposed models were focused on the procedure of the shipbuilding process, and were not concerned with the components of the multibody dynamics kernel. For this reason, the proposed elementary simulation model could be reused with a limitation, and its functionality was difficult to extend. Thus, this study focuses on the components of a multibody dynamics kernel such as bodies, forces, and wires, and tries to configure the components in a scenario in which they have one-to-one connection with the components of the multibody dynamics kernel. From this consideration, the component of the proposed scenario manager can be both reusable and extensible.

2.2. Discrete event simulation and discrete time simulation

Before explaining the detailed descriptions of the scenario management, we will shortly mention two kinds of simulation. To derive the process that status variables change by means of external and internal events, that is, to find the change of state variables according to the change of time is called *simulation*. A simulation that the state of a model changes by means of any events is called a *discrete event simulation*. The discrete event simulation processes the events, which change state variables of a model, in the order in which they occur. A simulation which calculates the state of a model every unit time is called a *discrete time simulation*. The discrete time simulation is being mostly used for analyzing dynamics or mechanics systems because it calculates the state of a model every unit time.

Zeigler et al. (2000) proposed formal structures, DEVS, and a discrete time system specification (DTSS), which can handle simulation models of a discrete event and time. They are widely used as standard formalisms of modeling and simulation. DEVS formalism is a hierarchical and modular modeling approach centered on the state concept. In its basic form, it does not consider the system structure evolution; only the states can evolve or move. Each system is described from the functional (behavioral) and structural aspects. Likewise, DEVS formalism is composed of two types of models: *atomic models* and *coupled models*. The atomic model represents the basic behavior of the system, and the computed model denotes its internal structure. On the other hand, the coupled model provides the method of assembly of several atomic and/or coupled models to build a complex system hierarchy.

The atomic models are the basic components of DEVS formalism. Their operation is close to the "state-machines". Formally, an atomic model of DEVS formalism is specified by 7 tuples:

$$\mathrm{M} = \langle X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta \rangle \tag{1}$$

In Eq. (1), X is the set of input event, Y is the set of output event, S is the set of sequential states, δ_{ext} is the external state function, δ_{int} is the internal transition function, λ is the output function, and ta is the time advanced function.

A coupled model is modular and presents a hierarchical structure, which allows the creation of complex models starting from atomic and/or coupled models. It is described as follows:

$$\mathrm{CM} = \langle X, Y, \{M_d\}, EIC, EOC, IC, select \rangle \tag{2}$$

In the coupled model, X is the set of input event, Y is the set of output events, and $\{M_d\}$ is the set of component models included in this coupled model. *EIC* (External Input Coupling), *EOC* (External Output Coupling), and *IC* (Internal Coupling) represent the relationship among its own external inputs, outputs, and those of child models. Additionally, *select* sets priorities of child models when the events fires simultaneously.

DTSS formalism is a model structure which continuously calculates the state of a model every unit time. An atomic model based on DTSS formalism has similar structure to that of the DEVS formalism, and in addition, it is connected with the atomic model of DEVS formalism. An overall system is composed of a set of component models, either atomic or coupled, thus being in a hierarchical structure. Each DEVS or DTSS model, also either atomic or coupled, has correspondence to an object in the real system to be modeled.

Bang developed a simulation framework based on the hybrid DEVS and DTSS formalism (Bang, 2006). To evaluate the efficiency and applicability of this simulation framework, it was applied to the block erection process in shipbuilding (Cha and Roh, 2010), the dive of a submarine (Ha et al., 2012a,b), and an analysis of the evacuation of a passenger ship (Ha et al., 2012a,b). DEVS formalism was also used to the simulation of subsea production systems (Woo et al., 2014). Most of analysis based on multibody system dynamics is time-based analysis. However, as aforementioned in previous section, the process of shipbuilding is a kind of event-based sequence. Thus, this study adopts the hybrid DEVS and DTSS formalism to cover both of time-based and event-based analysis. The simulation kernel developed by Bang was also used in this study.

3. Multibody dynamics simulator for the shipbuilding process

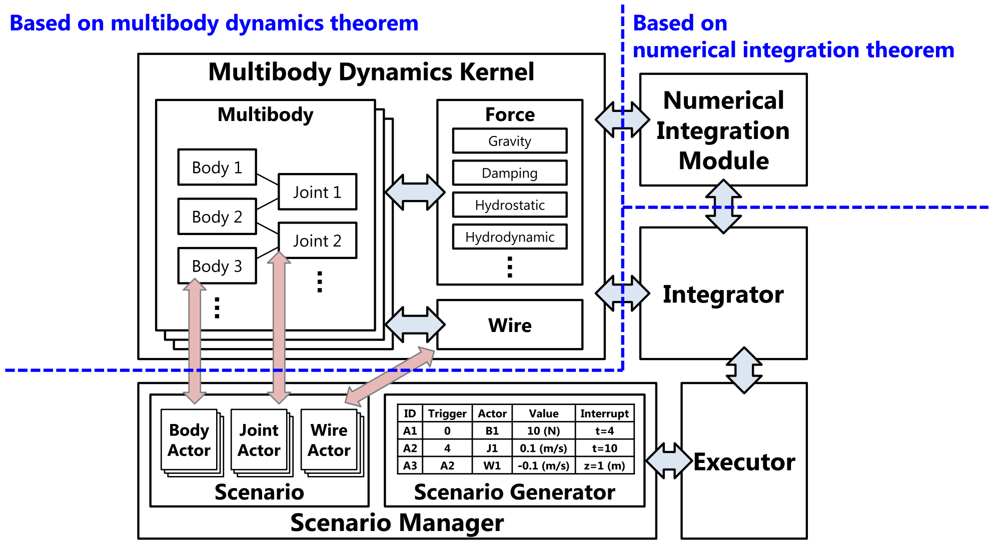
Before the scenario manager is described in detail, the developed multibody dynamics simulator for the shipbuilding process in previous study (Ku and Ha, 2014) is briefly previewed in this section. In Fig. 2, the multibody dynamics simulator has four components: the "Multibody Dynamics Kernel," "Numerical Integration Module," "Integrator," and "Scenario Generator."

For the modeling and dynamic analysis of the systems in shipbuilding simulation, a multibody dynamics kernel was developed based on the recursive Newton−Euler formulations (Shabana, 2005; Featherstone, 2008) in a previous study (Ku et al., 2012; Ku and Ha, 2014). The dynamics of a rigid-body system are described by its equation of motion, which specifies the relationship between the forces that act on the system and the accelerations they produce. The developed kernel contains the algorithms for the following two calculations: the calculation of the forward dynamics, or of the acceleration response of a given rigid-body system to a given applied force, and the calculation of the inverse dynamics, or of the force that must be applied to a given rigid-body system to produce a given acceleration response. The equations of motion for each body of a multibody system based on recursive formulation can be summarized as follows.

$$
\begin{aligned}
\widehat{\mathbf{v}}_i &= {}^i\mathbf{X}_{i-1}\cdot\widehat{\mathbf{v}}_{i-1}+\mathbf{S}_i\cdot\dot{q}_i,\ \widehat{\mathbf{a}}_i \\
&= {}^i\mathbf{X}_{i-1}\cdot\widehat{\mathbf{a}}_{i-1}+\mathbf{S}_i\cdot\ddot{q}_i+\dot{\mathbf{S}}_i\cdot\dot{q}_i+\widehat{\mathbf{v}}_i\times\mathbf{S}_i\cdot\dot{q}_i,\ \widehat{\mathbf{f}}_i^B \\
&= \widehat{\mathbf{I}}_i\cdot\widehat{\mathbf{a}}_i+\widehat{\mathbf{v}}_i\times{}^*\widehat{\mathbf{I}}_i\cdot\widehat{\mathbf{v}}_i,\ \widehat{\mathbf{f}}_i=\widehat{\mathbf{f}}_i^B+{}^i\mathrm{X}_{i+1}^*\cdot\widehat{\mathbf{f}}_{i+1}-\widehat{\mathbf{f}}_i^{ext}\quad \tau_i=\mathbf{S}_i^T\cdot\widehat{\mathbf{f}}_i
\end{aligned}
\tag{3}
$$

The kernel consists of multibodies, forces, and wires. A multibody represents each independent system such as a crane and a block. The kernel provides basic external forces for mechanical systems, such as gravitational force and damping force, and additional external forces, such as hydrostatic force and hydrodynamic force, for application to the shipbuilding process. To support the lifting sequence frequently used in shipbuilding, the kernel has a wire, so multibodies can be connected via wires.

To simulate and analyze the dynamic phenomena of the shipbuilding process for each time unit, a numerical integration module was developed in a previous study (Chat et al., 2010c). Because of some strict cases of the simulation of the shipbuilding procedure, the numerical integration module provides various integration methods such as the Euler method, Runge−Kutta method, Adams−Bashforth method, and Hilber−Hughes−Taylor method. Using the numerical integration module, the integrator integrates the equations of motion and calculates the position and velocity of each body in the given systems at each time unit. The scenario manager manages the simulation scenario and assigns the actors that act on the dynamics kernel according to the given scenario. The scenario manager is described in detail in the next chapter.



ID	Trigger	Actor	Value	Interrupt
A1	0	B1	10 (N)	t=4
A2	4	J1	0.1 (m/s)	t=10
A3	A2	W1	-0.1 (m/s)	z=1 (m)

Fig. 2. Configuration of multibody dynamics simulator for shipbuilding process (Ku and Ha, 2014).

4. Scenario manager for the multibody dynamics simulator

By analyzing a sequence of motions in multibody system dynamics, the simulation scenario was specified using two components: an "Action" and an "Action List." The Action is a user-defined input that acts on sub-components of multibody system dynamics, and the Action List is a specification of sequential or parallel actions. An "Actor" was introduced to represent each actions in the action list. A scenario was configured by combining and connecting these actors. In addition, a "Scenario Generator" was introduced to generate a scenario automatically according to the given action list. Since all actions are a kind of event, the actor was modeled based on DEVS formalism, generally used in event-based simulation. To conveniently connect with the actor, the scenario generator and an integrator, which is used to calculate an equation of motion based on multibody system dynamics, are also modeled based on DEVS formalism.

4.1. Specifications of the multibody dynamics simulation scenario

As briefly mentioned in Chapter 3, the target system for dynamic analysis is basically composed of multiple multi-bodies in multibody system dynamics. The multibodies interact with each other using wires or interact by collision. Each multibody consists of multiple bodies and joints. All the bodies in the multibody interact with other bodies through the connecting joints. A force can also act on each body. From these characteristics of multibody systems, the following two kinds of interaction can be specified:

(1) Interactions in a multibody, and
(2) Interactions between multibodies.

4.1.1. Interactions in a multibody

As mentioned in the previous paragraph, a multibody is composed of multiple bodies and joints. Fig. 3 shows an example of a two-link arm. The multibody in Fig. 3 is composed of three bodies: the base and two joints. Each joint is specified as a rotating joint.

There are two kinds of action for moving the given multi-body as the user desires: (1) with the forces that act on the bodies, and (2) by changing the position and velocity of the joints. For example, if the user wants to move the end point 'P1' of the body 'B2,' the proper force should act on the bodies 'B1' and 'B2,' or the position and velocity of the joints 'J1' and 'J2' should be changed properly. From these concepts, the interaction in a multibody was specified in this study as follows: (1) the component that acts on the bodies, i.e., the force, and (2) the component that acts on the joints, i.e., the position and velocity.

Actually, the changes in the position and velocity of the joint are not directly adjusted to the target joint. Using the change in the position and velocity of the joint, the internal controller module calculates the force that acts on the joint according to the PID (Proportional-Integral-Derivative) controller algorithm. To more intuitively set the inputs to the joints, the position and velocity of the joints were chosen, not the force that acted on the joints.

4.1.2. Interactions between multibodies

In a multibody dynamics system, each multibody moves independently. If one multibody acts on another multibody, they are considered one multibody, not two separate multi-bodies. The only case in which one multibody acts on another multibody is when they are connected via wires. Since the length of the wires is changed, the force that acts on each body, due to the wire tension, will change.

As shown in Fig. 4, the multibody "Crane" is composed of three bodies: the base and two joints; and the multibody "Block" is composed of only one body. These two multibodies are connected via a wire. To lift the block up or down, two actions can be considered: one that changes the pose of the multibody "Crane," and that which changes the length of the wire rope 'W1.' The change in a multibody was already mentioned as the interaction in a multibody, so the interaction between multibodies can be specified as follows: (1) the component that acts on the wires, i.e., the length; and (2) the speed of hoisting up or down.

4.2. Configuration of the scenario

In the previous section, the concept of each action was specified as interaction in a multibody and interaction between

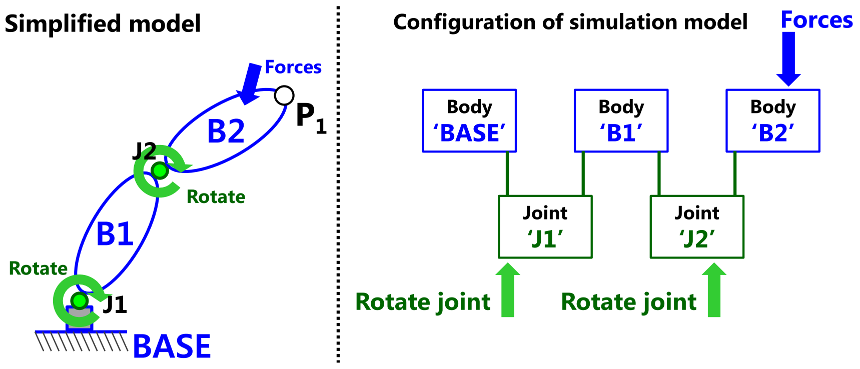


Fig. 3. Two-link arm as a simple example of multibody system.

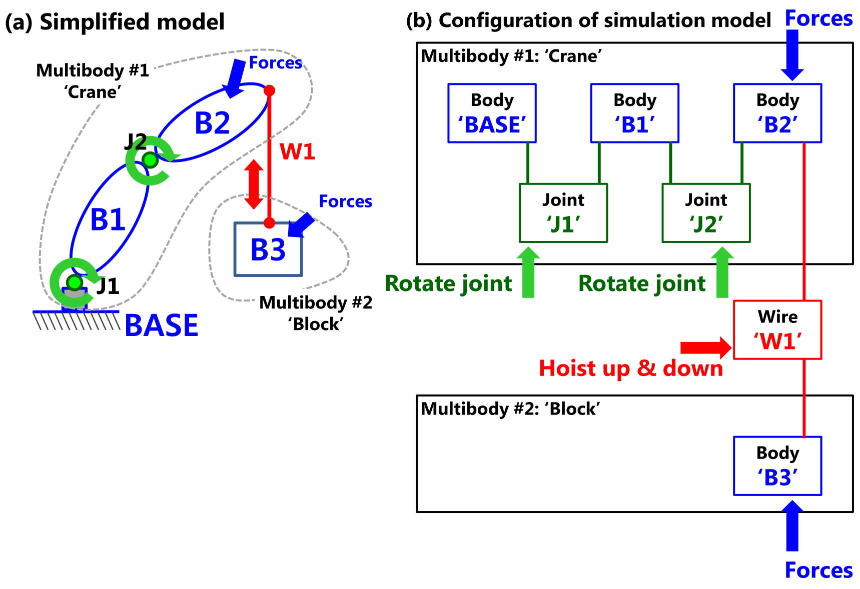


Fig. 4. Two-link arm and block connected via a wire: (a) simplified model of a crane and a block, and (b) configuration of a simulation model using the components of the multibody dynamics kernel.

multibodies. Each action should be performed at the proper time with the desired functions, so a model for performing this action, named "Actor," was defined. The scenario for the simulation of the shipbuilding process can be made according to the sequential and parallel compositions of such actions, or the so-called "Action List."

4.2.1. Actors — user-defined input that acts on multibodies

In this study, "Actor" means a user-defined input that acts on multibodies. All bodies, joints, and wires in multibody dynamics systems can be connected with multiple actors. According to the type of the interaction, the actor is specialized in the body, joint, and wire.

A body actor is an actor that acts on a body of a multibody. In a multibody system, only a force acts on a body, so the body actor has the function of acting as a specific force at a certain time. Actually, since a multibody is a kind of nonlinear system, it is very difficult to imagine how to change the position and velocity of the body when the specific force is acting on that body. Thus, the body actor is not well used to controlling the position and velocity of a certain body, and is used only to act as an environmental force such as a gravitational force, hydrostatic force, hydrodynamic force, wind force, or mooring force.

A joint actor is an actor that acts on a joint of a multibody. In a multibody system, a force also acts on the joint and changes its position and velocity. It is easier to imagine how to pose the joint when a force is acting on it than when a force is acting on a body, but the exact pose of the joint is still difficult to calculate. For more intuitive use of the joint actor, it allows the user to set a specific position and velocity of the joint as an input. As briefly mentioned in the previous section, the joint actor has a PID controller. The PID controller is a generic control loop feedback mechanism that is widely used in industrial control systems and can simply control the position or velocity of a certain joint by minimizing the error value as the difference between a measured process variable and a desired setpoint.

A wire actor is an actor that acts on a wire that connects two multibodies. The user can set the target length of the wire and the change in the wire length. After the length of the wire is changed, the tension force should act on the connected multibodies, so the wire actor has a module that calculates the tension force according to its length and elongation. The user can also set the type of wire as a typical wire or a spring.

Table 2 shows the list of basic actors used for the scenario management. Fig. 5 shows an example of using actors acting on each component of multibodies. Furthermore, there can be no interaction at a certain time for stabilizing the motion or for some other reasons. According to the research of Cha et al. (2010d), this action was considered an action of waiting, and was implemented by adding the waiting state to the actor model. However, adding a new state to a model is exhausting. Thus, in this study, the waiting action was implemented by adopting DEVS formalism and dynamic allocation of the actor models during the simulation. This concept will be detailed in the next section.

4.2.2. Action list — set of details of each procedure

Using the actors, the user might be able to take an action to an existing multibody system. However, to define the

Table 2
List of actors.

Actor	Item	Content
Body Actor	Target	Body
	Action	Force
Joint Actor	Target	Joint
	Action	Position or Velocity
Wire Actor	Target	Joint
	Action	Target Length and Velocity, and Wire Type

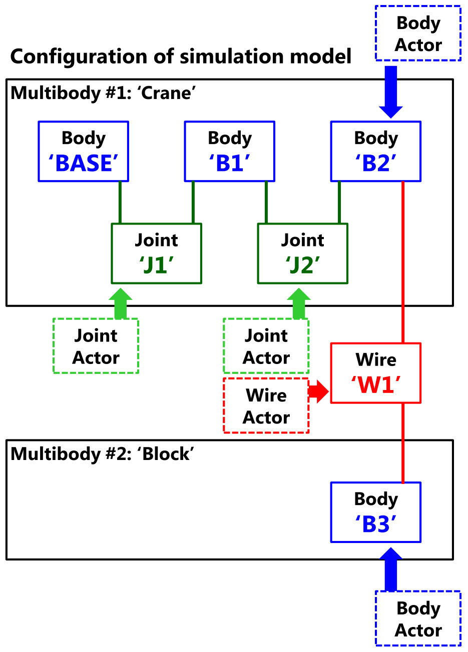


Fig. 5. Actors acting on multibody dynamics system.

procedure of the actions, it should be defined when the actor starts working and when it stops. The "Action List" is a set of details of each action in a simulation procedure, and lists how to run an existing multibody system according to a given objective.

Fig. 6 shows a sequence for generating a scenario for moving a block with a two-link arm using the actors. The multibody system is given in Fig. 4. The scenario for moving a block is as follows: (1) change the position and orientation of the "Crane," (2) hoist up the "Block," (3) change the position and orientation of the "Crane," and (4) hoist down the "Block."

In Step (1), to change the pose of the crane, two options were considered: acting out a specific force on the bodies of the crane using a body actor, and acting out a force on the joints of the crane using a joint actor. As mentioned in the previous section, the second option is a better choice, so two joint actors are allocated and connected with the two joints "J1" and "J2." These two actors should be started after the simulation starts, so the trigger for starting the action is given by the starting time, which is normally zero, to the first group of the actors. The end of the actions is checked based on the target position and orientation of the crane. After finishing the actions, the actors should send some messages to other actors to trigger the next actions. After Step (1) is finished, a wire actor will be allocated and connected to the wire "W1." To hoist up the block in Step (2), the length of the wire "W1" should be reduced according to the rate of change of the wire. This wire actor should be started after finishing Step (1), so the two joint actors should notice the end of the action in the next wire actor, and the wire actor will be triggered by certain events from the previous actors.

As mentioned in the previous paragraphs, to describe the simulation procedure, each action in an action list should have the following items: a trigger, the specifications of an action, and the end condition. The trigger is a condition for starting an action, such as a specific starting time or previous actions. The specifications of an action are the kind of actor and the initial values to be set, and the end condition is the condition for ending an action, given by the specific position and orientation of the joint actor, the specific length of the wire actor, and the specific time of all the actors.

Table 3 shows the detailed specifications of the actions in the action list for a block-lifting and transportation procedure. Each action has its own ID to identify it, and has specifications on how the actor is connected. The connected actor and the values for running it are noted in the specifications. For example, the action "A3" has the specifications "Actor: W1 and Value: $\dot{l} = -0.1$ m/min," which mean that the wire actor is connected to wire "W1" and will shorten the wire by 1 m per minute.

The action also has a trigger for starting to run the connected actor, and an end condition for stopping its running. The trigger has a type of start condition and the values according to this type. It also has an interval to delay the start of the action. For example, in the action "A3," the type of the trigger is "actor," the value is "A1 & A2," and the interval is set at 1 min. These mean that the wire actor that acts on wire "W1" will be started with a one-minute interval after the end of actions "A1" and "A2." As mentioned in the previous section, all the actions in the action list can have a default interval, so the actor need not add states to describe the waiting action. The end condition of action "A3" is also given

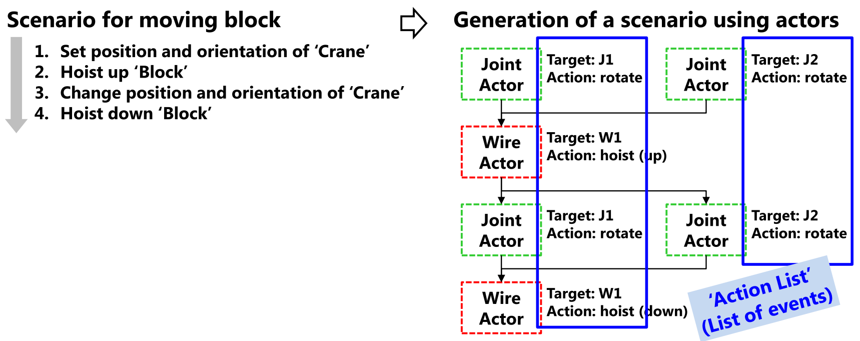


Fig. 6. Generation of a scenario using actors: scenario for moving a block using a two-link arm.

Table 3
Example of an action list for a block lifting and transportation procedure using a two-link arm.

ID	Trigger	Specification	End condition
A1	Type: time Value: 0 [min]	Actor: J1 Value: $\dot{\theta} = 0.1\ ^\circ/\text{min}$	$\theta = 75^\circ$
A2	Type: time Value: 0 [min]	Actor: J2 Value: $\dot{\theta} = 0.1\ ^\circ/\text{min}$	$\theta = -10^\circ$
A3	Type: actor Value: A1 & A2 Interval: 1 [min]	Actor: W1 Value: $\dot{l} = -0.1$ m/min	$l = 10$ m
A4	Type: actor Value: A3 Interval: 1 [min]	Actor: J1 Value: $\dot{\theta} = -0.1\ ^\circ/\text{min}$	$\theta = 70^\circ$
A5	Type: actor Value: A3 Interval: 1 [min]	Actor: J2 Value: $\dot{\theta} = -0.1\ ^\circ/\text{min}$	$\theta = -15^\circ$
A6	Type: actor Value: A4 & A5 Interval: 1 [min]	Actor: W1 Value: $\dot{l} = -0.1$ m/min	$l = 20$ m

models: a "Scenario," a "Scenario Generator," an "Integrator," and an "Executor." The Scenario model represents the sequence of the simulation, and it modeled as a coupled model according to the DEVS formalism. The user can make the Scenario model by combining and connecting various types of actor using the graphic user interface (GUI). The Scenario Generator model is an atomic model which can automatically generate a Scenario model according to the given action list. The Integrator model is also an atomic model, and it is used to integrate the equations of motion of target system based on multibody system dynamics. The Executor has the functions to control the simulation. It can start, pause, stop, reset and replay the simulation. At the start of the simulation, it triggers the scenario generator, the scenario, and the integrator sequentially (see Fig. 7).

The detailed specification of overall model is as follows. In the following sections, each sub-model will be described in more detail.

Specification of overall coupled model

$CM_{Overall} = < X, Y, \{M_i\}, EIC, EOC, IC, select >$
X={"SimulationCtrl" }
Y={"Result"}
$\{M_i\}$={Scenario, Scenario Generator, Executor, Integrator}
EIC: (SimulationCtrl, Executor.Control),
(SimulationCtrl, Scenario Generator.Trigger)
EOC: (Executor.Result, Result),
(Integrator.Result, Result),
(Scenario.Result, Result)
IC: (Scenario Generator.Action, Scenario.Action),
(Executor.StartScenario, Scenario.Start),
(Executor.StartIntegration, Integrator.Trigger)
(Executor.Stop, Integrator.Trigger)
$select$: Scenario Generator, Executor, Scenario, Integrator (first in higher priority)

by "$l = 10$ m," which means the actor will stop the action when the length of wire "W1" is set at 10 m.

4.3. Model design for scenario management using DEVS formalism

According to the specifications and configuration of the scenario, we developed the models for the scenario management. The models represent abstract behavior of each components for the scenario management, such as the actor for the description of each action, the integration algorithm for the numerical integration of the equation, and the execution of the simulation, using the DEVS formalism. The proposed model provides the user with a generic model structure for the scenario management, by means of which a detailed behavioral description relating to the simulation based on multibody system dynamics can be conducted. Each model is described in detail in the next sections.

4.3.1. Overall model structure

The developed overall model structure is described in Fig. 1. The overall model structure consists of four main

4.3.2. Scenario and actor model

The "Scenario" model is composed of various "Actor" models. The "Actor" atomic model focuses on a common action of the actors such as a body actor, a joint actor, or a wire actor. The actor model takes on the role of acting external force to the target body, joint, or wire. A sequential or parallel jobs in shipyard can be modeled by combining and connecting these actors. The major roles of the actor models are as follows:

- waiting: wait for trigger to start the action (the start signal will be sent by previous actors or the user);
- acting: act external force to the target body, joint, or wire;
- checking: check the end condition to finish the given job up.

The following specification represents DEVS-based actor model in detail. Fig. 8 shows the state transition diagram of the Actor atomic model based on DEVS formalism. In Fig. 8, the circles are the states of the model, and the double-lined circle means the initial state of the model. Since the set-theoretic specification can be easily turned into the diagram, we will use this diagram to show DEVS atomic model of this paper.

Specification of "Actor" atomic model

$AM_{Actor} = < X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta >$
X={"Trigger 1", "Trigger 2", … , "Interrupt"}
Y={"Done"}
S={"READY", "WAIT", "ACT", "CHECK", "SEND", "END"}

δ_{ext}:	READY × All X → WAIT
	ACT × "Interrupt" → SEND
	CHECK × "Interrupt" → SEND
δ_{int}:	WAIT → ACT
	ACT → CHECK (execute two functions: *Act*(), Finished = *Check*())
	CHECK → ACT (if Finished == false)
	CHECK → SEND (if Finished == true)
λ:	SEND → "Done" (if Finished == true)
ta:	READY = ∞
	WAIT = t_1 (Waiting Time)
	ACT = Δt (Unit Time Step)
	CHECK = 0
	SEND = 0
	END = ∞

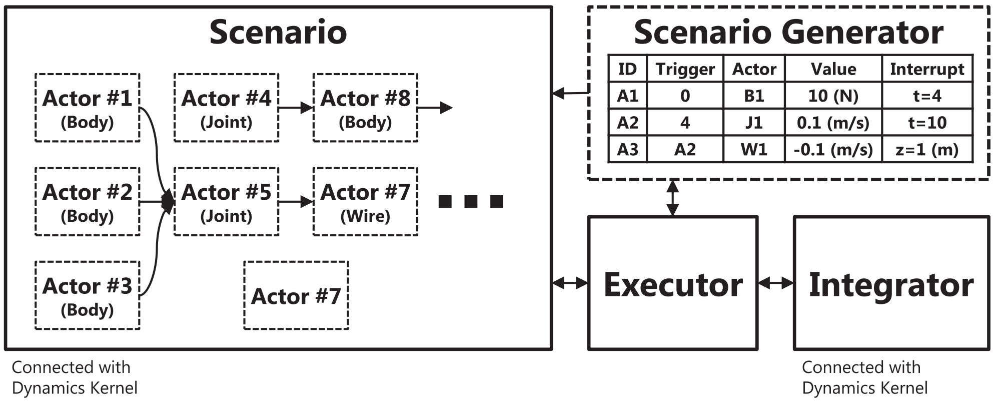
In DEVS-based specification of the actor atomic model, t_1 means the waiting time to start the action after getting the trigger, and Δt represents the unit time step. As described in Section 4.2, all actors have a trigger, an end condition, and specifications. The trigger is specified as the input event "Trigger #." The READY state is also concerned with the start of each action. If all trigger events are fired, the state will be changed to the WAIT state. After waiting for the given interval at the WAIT state, the state will be changed to the ACT state and will start running the action at each time unit Δt by going to and coming from the CHECK state. The end condition of each action is checked in the CHECK state. If the actor satisfies the end condition or is interrupted by the other input event "Interrupt," it will stop running and will change its own state to the END state.

Two functions with the name of "Act" and "Check" are executed when the state is changed from CHECK to ACT. These two functions are inherited and overridden according to the characteristics of each actor such as body, joint, and wire. By inheriting this common actor atomic model, a body, joint, and wire actor atomic model can be configured. All their states, events, and transition functions are the same as those in the common actor atomic model, and only the functions "Act" and "Check" are changed to fit into the function of each actor. These mean the user can add a user-defined actor atomic model by inheriting the common actor atomic model and without changes in the configuration of the actor atomic model, if the user needs an additional actor that has different functions from those of the existing actors.

4.3.3. Scenario generator model

The "Scenario Generator" atomic model focuses on generating the scenario model according to the actions in the given action list. The scenario generator atomic model has two states: IDLE and GEN. At the start of the simulation, the "Executor" model send a trigger to this model through the "Start" input port. From receiving this trigger, the state of this model will be changed to the GEN state. At the GEN state, the model calls the function "to generate an actor according to the given action in the action list." If all actions in the action list is already generated to the concerned actors, it means that the scenario atomic model is configured well according to the given action list, so the scenario generator atomic model changes its own state to the IDLE state.

Fig. 9 shows the state transition diagram of the Actor atomic model based on DEVS formalism. The following specification represents DEVS-based actor model in detail.



ID	Trigger	Actor	Value	Interrupt
A1	0	B1	10 (N)	t=4
A2	4	J1	0.1 (m/s)	t=10
A3	A2	W1	-0.1 (m/s)	z=1 (m)

Fig. 7. Overall model structure for the scenario management.

Specification of "Scenario Generator" atomic model

$AM_{ScnearioGenerator} = < X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta >$
X={"Start" }
Y={"Action"}
S={"IDLE", "GEN" }
δ_{ext}: IDLE × "Start" → GEN
δ_{int}: GEN → GEN : if Finished == false
then execute two functions: Action = *GenerateAction*(), Finished = *IsFinished*()
GEN → IDLE : if Finished == true
λ: GEN → "Action" : if Finished == false
ta: IDLE = ∞
GEN = 0

4.3.4. Executor model

The "Executor" model has the functions for the control of whole simulation. In the idle state, this model will get the signal from the user directly through the "Control" input port, then the simulation will start. At first, this model sends a signal to the "Scenario Generator" model in order to generate the "Scenario" model according the given action list. After finishing the generation of the scenario, it will trigger the "Scenario" model to run the shipbuilding process simulation. In addition, it will also initiate the "Integrator" model to calculate the motions of each sub-component in target multibody systems such as cranes, blocks, and wires.

This model also has some functions to control the simulation. It means that this model can pause, replay, and stop the simulation. When the user sends some signal to control the

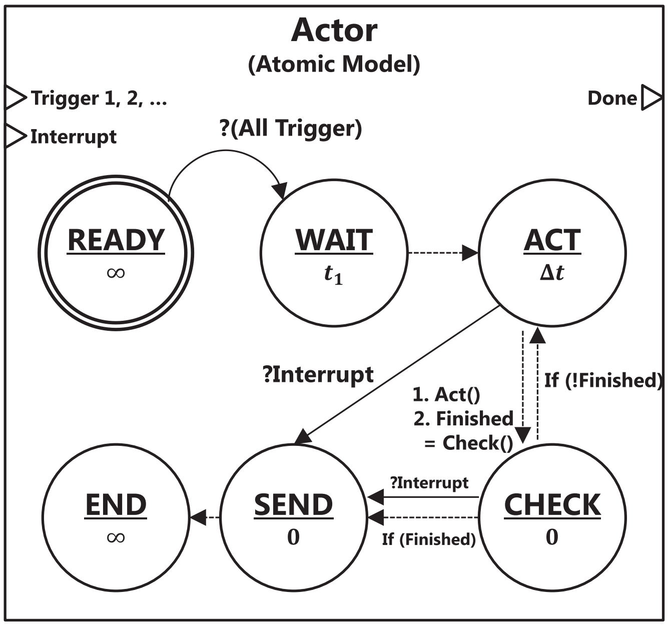


Fig. 8. Generic "Actor" atomic model based on DEVS formalism.

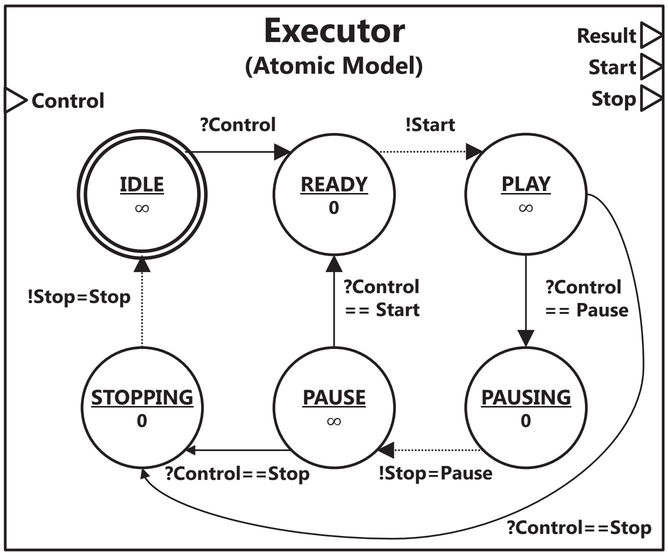


Fig. 10. "Executor" atomic model based on DEVS formalism.

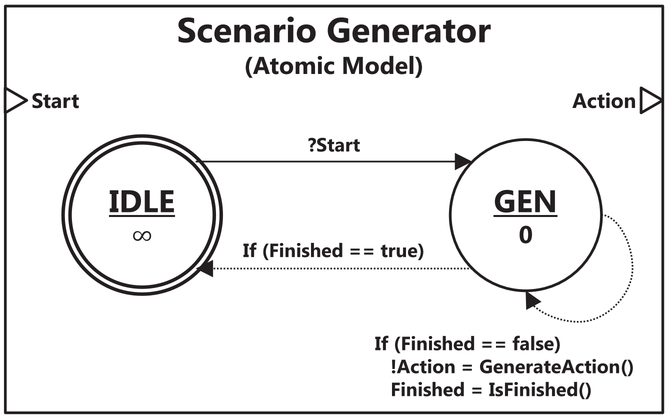


Fig. 9. "Scenario Generator" atomic model based on DEVS formalism.

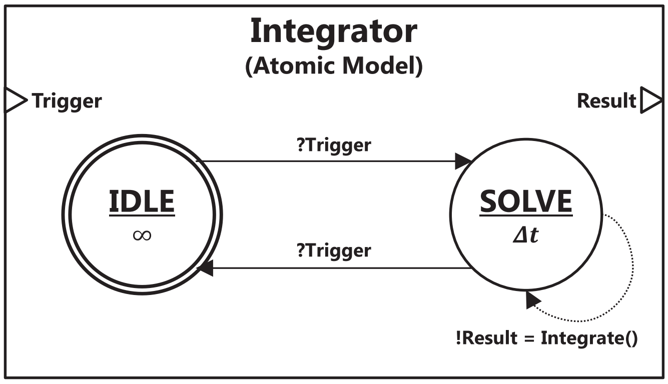


Fig. 11. "Integrator" atomic model based on DEVS formalism.

simulation, this model will change its own states and also send some signals to other models connected. Fig. 10 shows the state transition diagram of the Integrator atomic model based on DEVS formalism. The following specification represents DEVS-based actor model in detail.

Specification of "Executor" atomic model

$AM_{Executor} = < X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta >$
X={"Control"}
Y={"Result", "Start", "Stop"}
S={"IDLE", "READY", "PLAY", "PAUSING", "PAUSE", "STOPPING"}

δ_{ext}:	IDLE × "Control" → READY
	PLAY × "Control" → PAUSING (if Control == Pause)
	PLAY × "Control" → STOPPING (if Control == Stop)
	PAUSE × "Control" → READY (if Control == Start)
	PAUSE × "Control" → STOPPING (if Control == Stop)
δ_{int}:	READY → PLAY
	PAUSING → PAUSE
	STOPPING → IDLE
λ:	READY → "Start"
	PAUSING → "Stop"
	STOPPING → "Stop"
ta:	IDLE = ∞
	READY = 0
	PLAY = ∞
	PAUSING = 0
	PAUSE = ∞
	STOPPING = 0

4.3.5. Integrator model

For the dynamic analysis of shipbuilding process, it is necessary to solve the equations of motion based on multibody system dynamics every unit time. The "Integrator" model has the functions for numerical integration of the equations of motion. As shown in Fig. 11, this model is very simple and it only performs the numerical integration when the simulation starts. The user can surely choose a method among various integration methods such as Runge−Kutta method, Adams−Bashforth method, Hilber−Hughes−Taylor method, etc.

Fig. 11 shows the state transition diagram of the Integrator atomic model based on DEVS formalism. The following specification represents DEVS-based actor model in detail.

Specification of "Integrator" atomic model

$AM_{Integrator} = < X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta >$
X={"Trigger" }
Y={"Result"}
S={"IDLE", "SOLVE" }

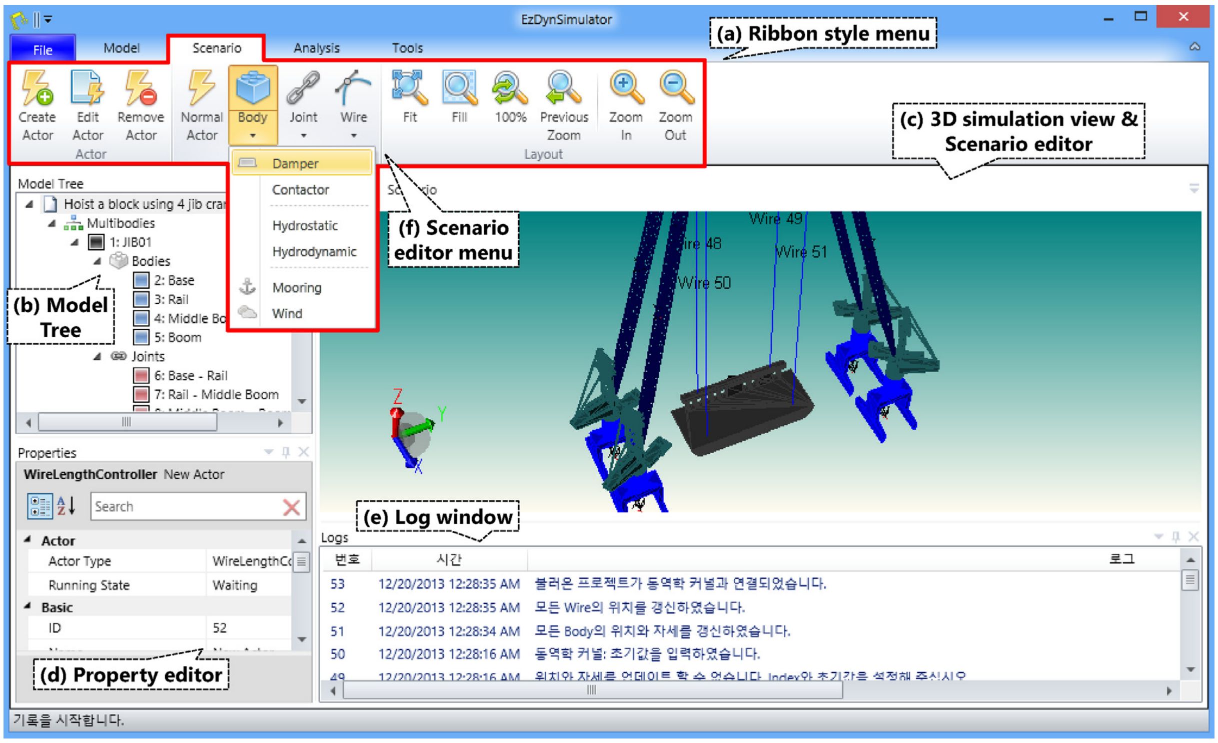
δ_{ext}:	IDLE × "Trigger" → SOLVE
	SOLVE × "Trigger" → IDLE
δ_{int}:	SOLVE → SOLVE : Result = *Integrate*()
λ:	SOLVE → "Result"
ta:	IDLE = ∞
	SOLVE = 0

4.4. Model implementation

Fig. 12 shows the developed multibody dynamics simulator with the proposed scenario manager. As shown in Fig. 12(a)−(e), the graphic user interface (GUI) of the developed simulator has six components: a ribbon-style menu, model tree, property editor, three-dimensional simulation view, scenario editor, and log window.

Fig. 12(f) shows the detailed user interface of the developed scenario manager. As mentioned in the previous sections, three categories of actors are proposed in this study: a body actor, a joint actor, and a wire actor. To start multiple actors simultaneously with only one trigger and considering the waiting time, an additional actor, called a 'Normal Actor,' was also implemented in the scenario editor, as shown in Fig. 12(f). The body actor, joint actor, and wire actor have various subtypes to support various types of external forces and interconnections between multibodies. For example, the body actor has six subtypes: Damper, Contactor, Hydrostatic, Hydrodynamic, Mooring, and Wind. The body actor of the 'Damper' type has a module for calculating the damping force, so the damping force can be acted on the body by executing this actor. Similarly, the body actor of the 'Hydrodynamic' type has a module for calculating the linearized hydrodynamic force, so the hydrodynamic force can also be acted on the body using this actor, and is a unique item of the shipbuilding process − rarely supported in other multibody dynamics simulators.

Fig. 13 shows the user interface of the developed scenario manager and an example of the block-lifting sequence



Actor	
Actor Type	WireLengthC
Running State	Waiting
Basic	
ID	52

```
번호  시간                    로그
53    12/20/2013 12:28:35 AM  불러온 프로젝트가 동역학 커널과 연결되었습니다.
52    12/20/2013 12:28:35 AM  모든 Wire의 위치를 갱신하였습니다.
51    12/20/2013 12:28:34 AM  모든 Body의 위치와 자세를 갱신하였습니다.
50    12/20/2013 12:28:16 AM  동역학 커널: 초기값을 입력하였습니다.
49    12/20/2013 12:28:16 AM  위치와 자세를 업데이트 할 수 없습니다. Index와 초기값을 설정해 주십시오
```

Fig. 12. The multibody dynamics simulator developed in this study: (a) ribbon style menu, (b) model tree of the target multibody systems, (c) three-dimensional simulation view and scenario editor view, (d) property editor to edit the properties of bodies, joints, etc., (e) log window, and (f) scenario editor menu in the ribbon style menu.

scenario using four jib cranes shown in Fig. 12(c). In Section 4.3, the models for the scenario management were mentioned as follows; the scenario, actors, scenario generator, executor, and integrator model. All models are implemented in this study, but only the actor models are shown to the user through the GUI. The user can create and modify the scenario by configuring the actors, and the other models are used in the background of the developed simulator. All actors are visualized as a rounded box, and the sequence of a scenario can be configured by connecting these actor boxes. All actors in the same category have the same color. The 'Collector' actor in Fig. 13 is a kind of 'Normal Actor,' and is a dummy actor that simultaneously collects the signal for ending previous actors and triggers the following actors.

5. Applications

To confirm the flexibility and usefulness of the developed scenario manager, it was applied to various examples in shipbuilding process. In this paper, the following examples of its applications are introduced:

- double compound pendulum,
- block-lifting, transportation and turn-over simulation by using two goliath cranes, and
- lifting and transporting a mega-block by using a floating crane and two goliath cranes.

5.1. Double compound pendulum

5.1.1. Outline

To verify the functions of the scenario manager, it was applied to simple example, double compound pendulum, with simple scenario. Fig. 14(a) shows the configuration of double compound pendulum. Any swinging rigid body free to rotate about a fixed horizontal axis is called a compound pendulum. As shown in Fig. 14(a), Body 0, 1 and 2 are rigid bodies, and Body 1 is interconnected to Body 0 and Body 2 by rotational joints. Body 0 is fixed to the inertial space and does not move.

5.1.2. Configuration of scenario

At the rest of double compound pendulum, its subcomponents, rigid bodies, do not move. Thus, it is tried to act external forces to its joints step by step. At first, we changed the angle of Joint 0—45° and the angle of Joint 1—30° simultaneously. After then, we rotated Joint 0 additional 90°.

These two actions can be represented to the scenario by using the joint actor. Fig. 14(b) shows this scenario represented by multiple joint actors. When the simulation starts, Joint Actor 1 and 2 in Fig. 14(b) will be activated, so the external forces will act on the target joints — Joint 0 and 1. These actors will be deactivated when Joint 0 reaches to 45° and Joint 1 reaches to 30°. When each joint finished its job, it will send a trigger to the connected joint, Joint Actor 3 in this example. Since Joint Actor 3 gets two triggers from Joint Actor 1 and 2, it will be activated finally and act on an

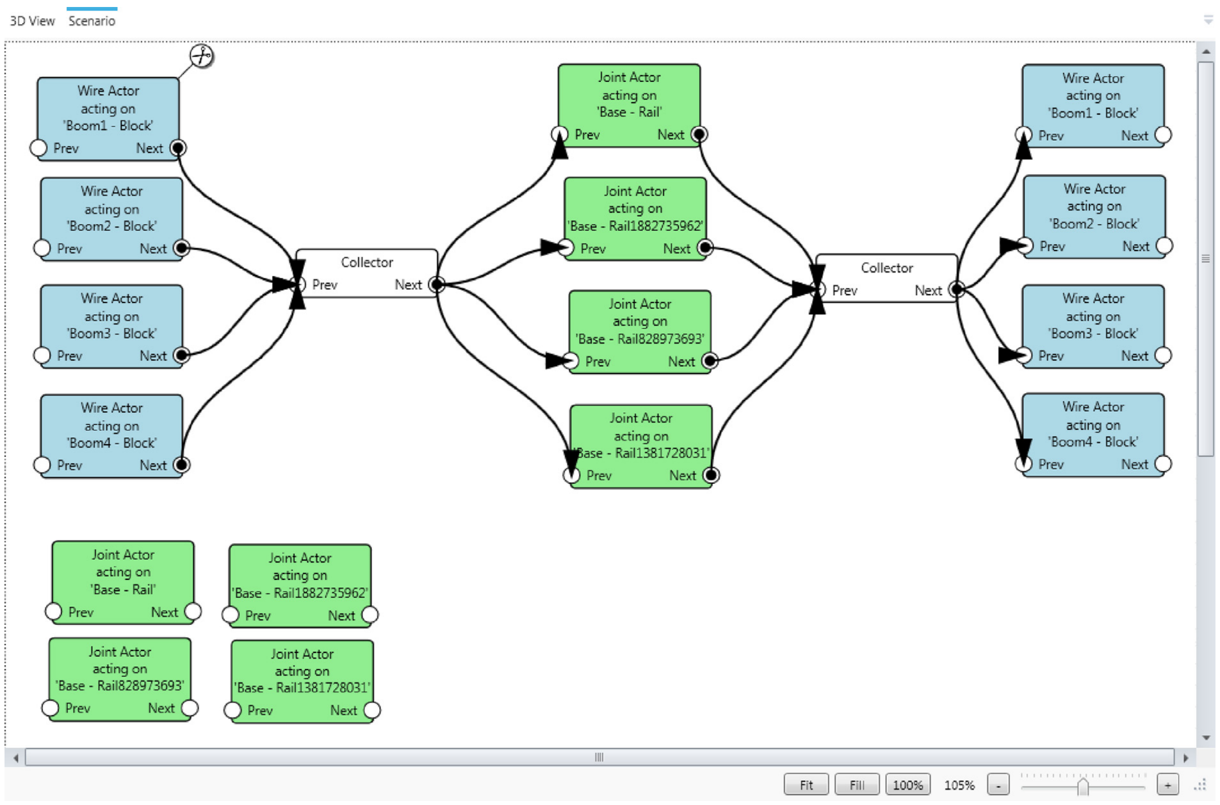

3D View
Scenario
Wire Actor acting on 'Boom1 - Block' Prev Next
Wire Actor acting on 'Boom2 - Block' Prev Next
Wire Actor acting on 'Boom3 - Block' Prev Next
Wire Actor acting on 'Boom4 - Block' Prev Next
Collector Prev Next
Joint Actor acting on 'Base - Rail' Prev Next
Joint Actor acting on 'Base - Rail1882735962' Prev Next
Joint Actor acting on 'Base - Rail828973693' Prev Next
Joint Actor acting on 'Base - Rail1381728031' Prev Next
Collector Prev Next
Fit
Fill
100%
105%

Fig. 13. User interface of the scenario manager for the shipbuilding process.

external force to the connected joint (Joint 0) to rotate it additional 90°.

Fig. 15 shows the configuration of double compound pendulum and the scenario configuration using the developed scenario manager. Fig. 15(a) shows that three rigid bodies are modeled and they are interconnected using two rotating joints. In Fig. 15(b), three joint actors are used to represent the given scenario. Two actors on the left will be activated simultaneous just after the simulation starts, and the actor on the right will be activated after two left actors are finished their jobs.

5.1.3. Simulation results

Figs. 16 and 17 shows the simulation result of double compound pendulum with the given scenario mentioned in previous section. As shown in Figs. 16 and 17 the change of Euler angles of each bodies is divided by four steps. The simulation started, and Actor 1 and 2 was activated simultaneously. Actor 1 tried to change the degree of Joint 0—45°, and Actor 2 tried to change the degree of Joint 1—30°. Since we have set that those angular velocities are the same, it was obvious that Actor 2 did its job faster than Actor 1. Thus, during from about 5 to 16 s after the simulation started, Actor 1 was still activated, but Actor 2 was deactivated after 5 s.

After about 16 s, Actor 1 was also deactivated, and then Actor 3 was activated. Actor 3 tried to change the angle of Joint 0, so both angles of Joint 0 and 1 were changed. After rotating Joint 0 additional 90°, Actor 3 was deactivated, and the execution of scenario was finished at about 25 s.

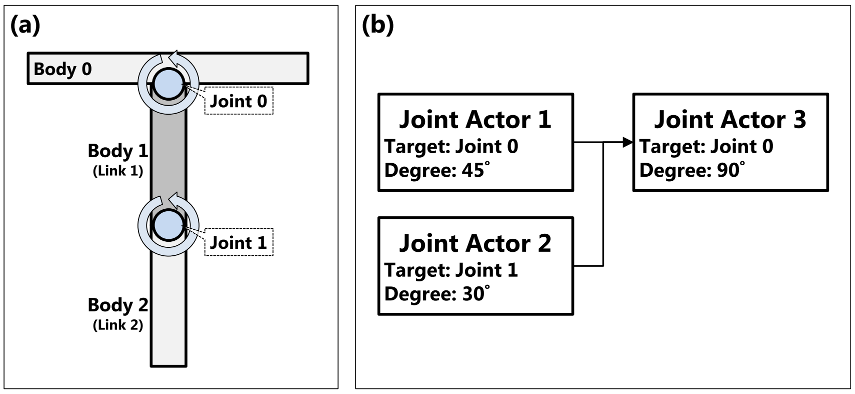


Fig. 14. Double compound pendulum: (a) system configuration, (b) scenario configuration.

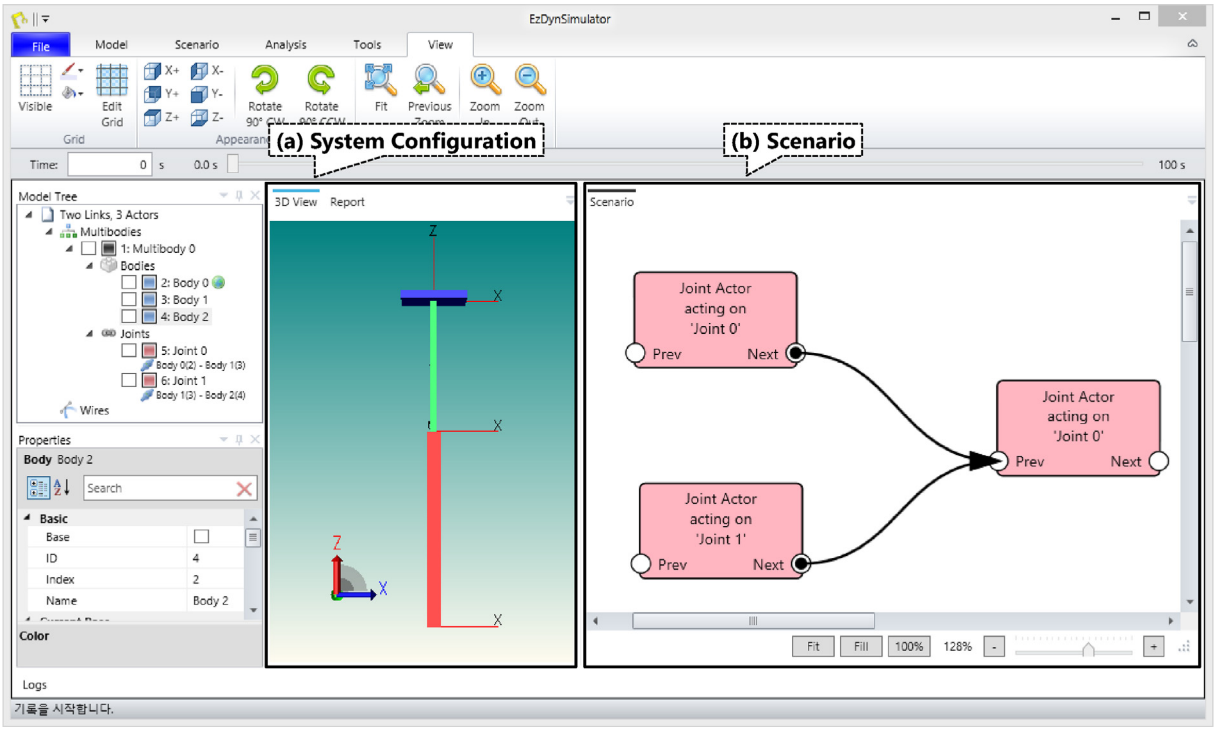


Fig. 15. Configuration of double compound pendulum system and scenario configuration using the developed program.

5.2. Block-lifting, transportation and turn-over simulation by using two goliath cranes

5.2.1. Outline

The block-lifting, transportation, and turnover procedure is the most frequent production process in shipyards. Fig. 18 shows an example of a block-lifting, transportation, and turnover process using two goliath cranes in a shipyard. The block with a weight of about 800 tons is connected to two goliath cranes via the wires. Some parts of these wires are grouped by a block loader.

Fig. 18(b) shows a sequence of the operation for the block-lifting, transportation, and turnover process. At first, two goliath cranes lift the mega block by shortening the connected wires. Then the two goliath cranes move to a certain location to move the block into the dock. Finally, the goliath cranes continue the turnover process by alternately shortening and lengthening each wire.

To simulate this operation sequence, dynamics models, simulation models that include actors, and the scenario should be defined. The detailed descriptions of the configuration of the simulation will be explained in the following sections.

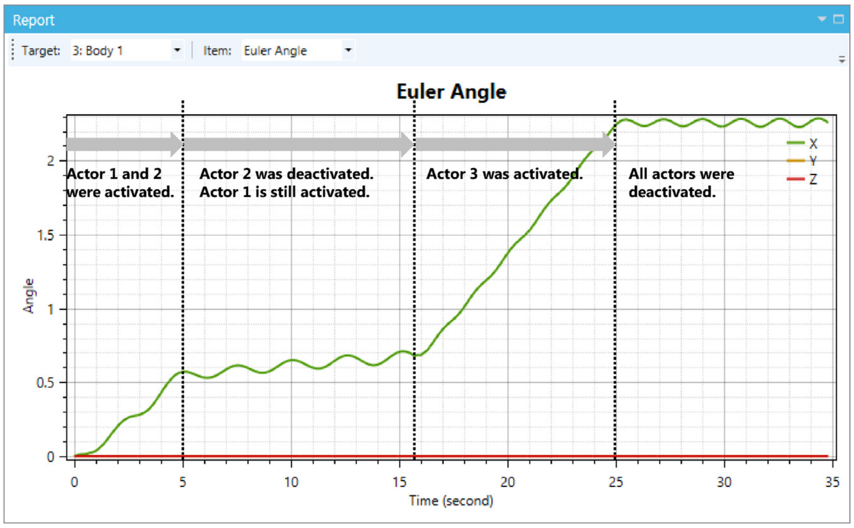


Fig. 16. Simulation result: change of Euler angles — Body 1 of double compound pendulum.

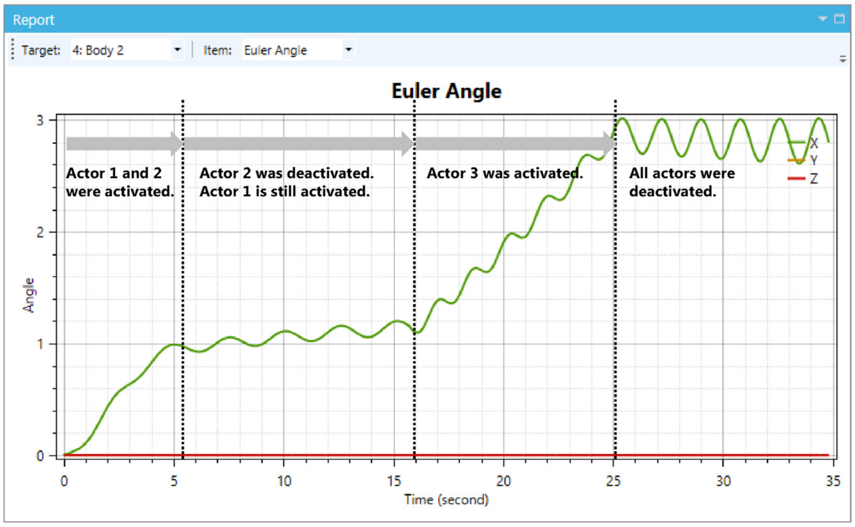


Fig. 17. Simulation result: change of Euler angles — Body 2 of double compound pendulum.

5.2.2. Model configuration of the multibody dynamics system

To analyze the dynamic responses through the simulation, the block-lifting and transportation are carried out using two goliath cranes, six block loaders, and one block, as shown in Fig. 18(a). The goliath crane is composed of the main body, the upper trolley, and the lower trolley. The upper trolley and the lower trolley are interconnected with the main body by sliding joints. The block loader consists of two bodies interconnected by a revolute joint. To simulate the block-lifting and transport, dynamics models should be made using the developed multibody dynamics kernel.

5.2.3. Scenario configuration

Fig. 19 shows the entire configuration, including the dynamics models, simulation models, and scenario, for the block-lifting and transportation simulation. The actor atomic models are allocated to some dynamic models that need to act according to the action list in the scenario. For example, a joint connects the body “Upper Trolley” and the body “Body,” and the joint actor atomic model is allocated to this actor.

In the right box in Fig. 19, the block-lifting and transportation sequence scenario is specified using an action list. Various actions are defined in the action list, including an action for moving the crane and an action for lifting the block up and down.

Fig. 20 shows the scenario configuration of target simulation using the developed scenario manager. Since the target scenario has many operations for hoisting wire up and down, many wire actors are exist in the scenario as shown in Fig. 20. At first, eight wire actors were activated and tried to hoist the block up after the simulation started (Fig. 20(a)). Then two joint actors tried to move two goliath cranes to the given position for turn-over and erection (Fig. 20(b)). Later, eight wire actors and two joint actors for lower trolley tried to turn the block over (Fig. 20(c)).

5.2.4. Simulation results

Fig. 21 shows the simulation results. The graph shows the change in the tension that acted on Block Loader 1. The results show that the scenario was successfully progressed by activating and deactivating the actors. At first, the wire actor

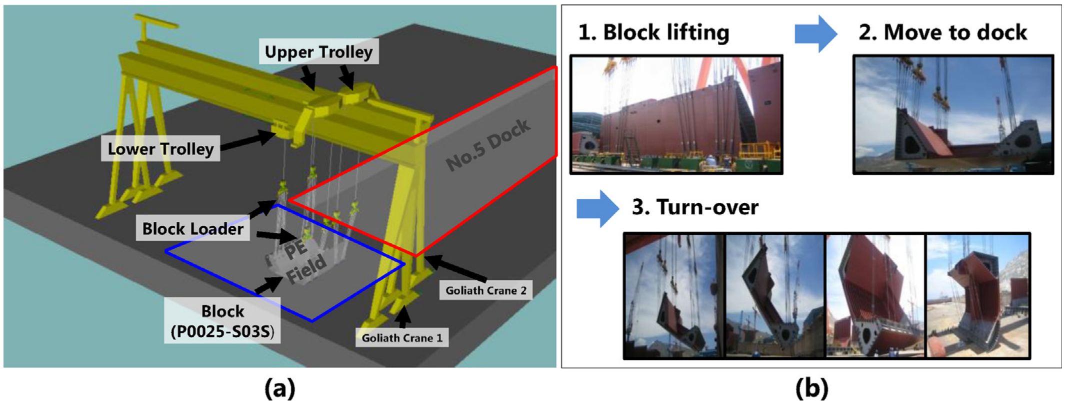


Fig. 18. Two goliath cranes and a block for the block-lifting, transportation, and turnover procedure: (a) conceptual modeling result of the simulation and (b) operation sequence.

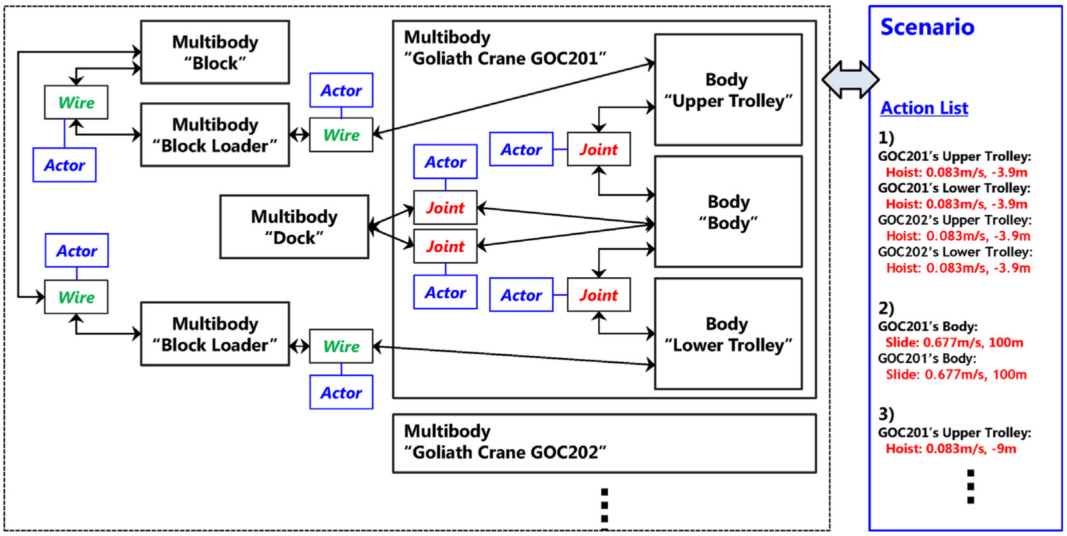
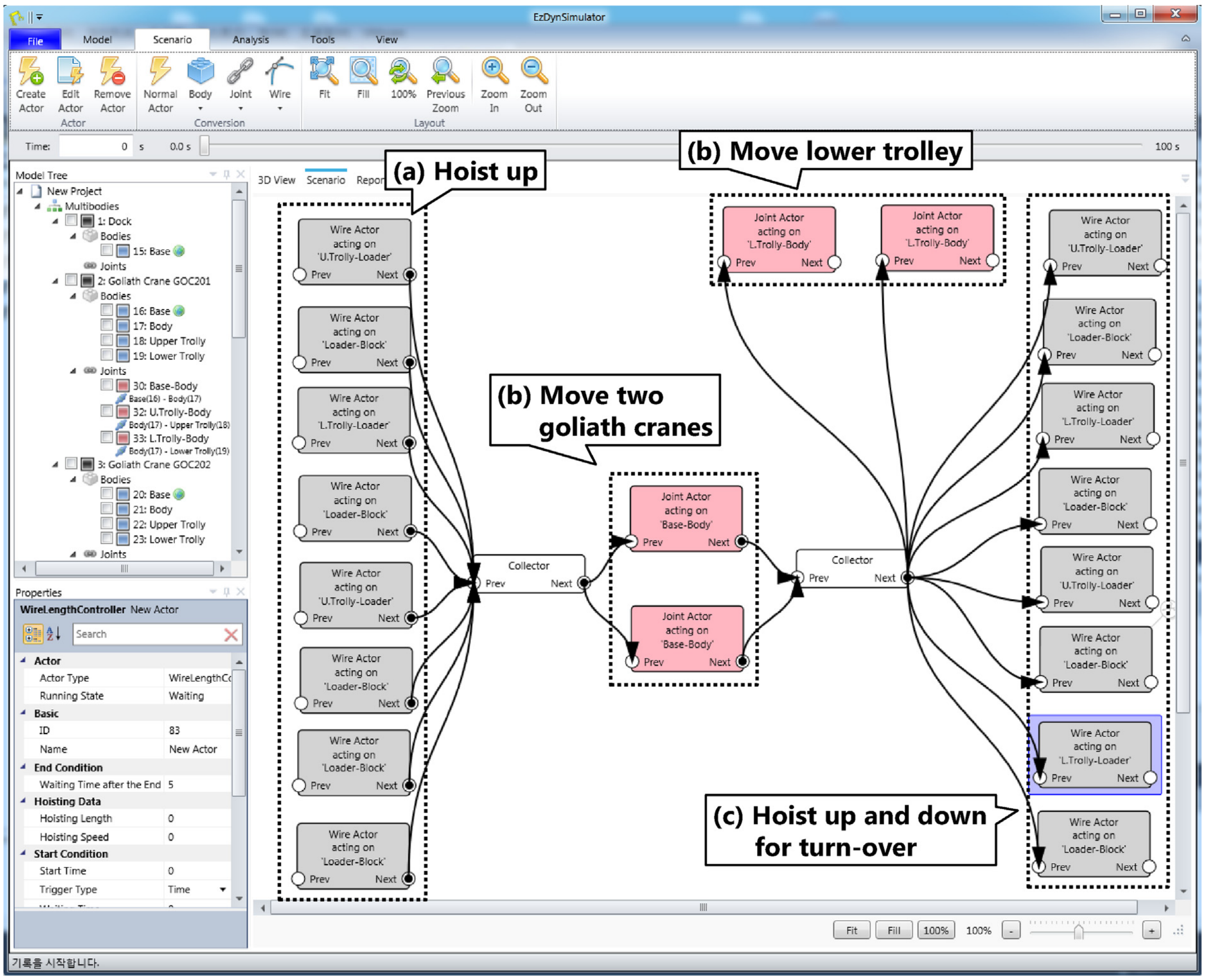

Multibody "Block"
Wire
Actor
Multibody "Block Loader"
Actor
Wire
Multibody "Goliath Crane GOC201"
Body "Upper Trolley"
Actor
Joint
Actor
Joint
Joint
Body "Body"
Multibody "Dock"
Actor
Actor
Joint
Body "Lower Trolley"
Actor
Wire
Multibody "Block Loader"
Wire
Actor
Multibody "Goliath Crane GOC202"
Scenario
Action List
1)
GOC201's Upper Trolley: Hoist: 0.083m/s, -3.9m
GOC201's Lower Trolley: Hoist: 0.083m/s, -3.9m
GOC202's Upper Trolley: Hoist: 0.083m/s, -3.9m
GOC202's Lower Trolley: Hoist: 0.083m/s, -3.9m
2)
GOC201's Body: Slide: 0.677m/s, 100m
GOC201's Body: Slide: 0.677m/s, 100m
3)
GOC201's Upper Trolley: Hoist: 0.083m/s, -9m

Fig. 19. Dynamics models, simulation models, and scenario for the block-lifting and transportation simulation.

tried to change the length of the target wire, so the graph shows that the tension acting on the wire increased. After the block-lifting ended, the goliath crane started to move to the dock, and it was performed by activating the joint actors on the goliath crane. After then, the goliath carne stopped moving and the block turn-over started. This procedure can be confirmed with the change of the tension acting on the wire in Fig. 21.



Model Tree
- New Project
  - Multibodies
    - 1: Dock
      - Bodies
        - 15: Base
      - Joints
    - 2: Goliath Crane GOC201
      - Bodies
        - 16: Base
        - 17: Body
        - 18: Upper Trolly
        - 19: Lower Trolly
      - Joints
        - 30: Base-Body
          - Base(16) - Body(17)
        - 32: U.Trolly-Body
          - Body(17) - Upper Trolly(18)
        - 33: L.Trolly-Body
          - Body(17) - Lower Trolly(19)
    - 3: Goliath Crane GOC202
      - Bodies
        - 20: Base
        - 21: Body
        - 22: Upper Trolly
        - 23: Lower Trolly
      - Joints

Properties
WireLengthController New Actor

Actor	
Actor Type	WireLengthC
Running State	Waiting
Basic	
ID	83
Name	New Actor
End Condition	
Waiting Time after the End	5
Hoisting Data	
Hoisting Length	0
Hoisting Speed	0
Start Condition	
Start Time	0
Trigger Type	Time

Fig. 20. Block-lifting, transportation and turn-over simulation by using two goliath cranes: scenario configuration using the developed program.

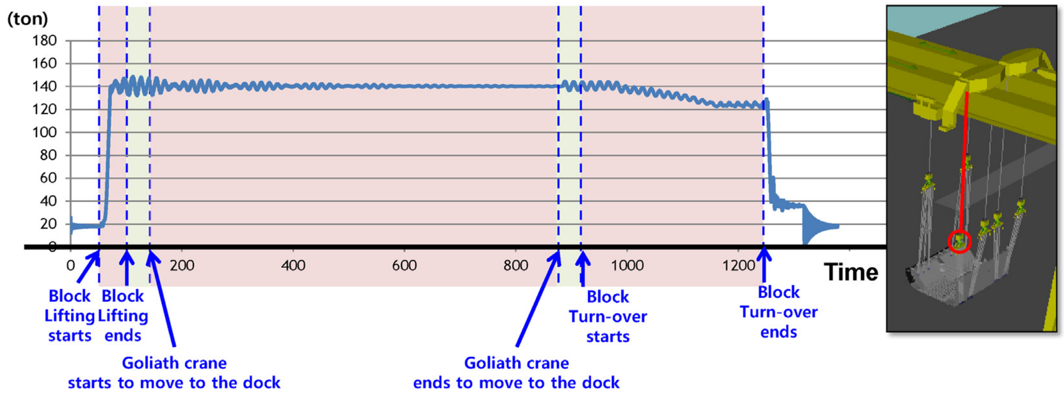


Fig. 21. Tension of the block loader 1 calculated in the block-lifting, transportation, and turnover simulation using the developed program.

5.3. Lifting and transporting a mega-block by using a floating crane and two goliath cranes

5.3.1. Outline

Fig. 22 shows an example of the application for lifting and transporting a mega-block by using a floating crane and two goliath cranes. In shipyard, there are many mega-block above 1000 tons to lift up and down, a floating crane and goliath crane have their limited capacities, so they are often used to co-work with other cranes. As shown in Fig. 22(a), one floating crane and two goliath cranes are tried to lift and move the target block with the weight of 3500 tons. Fig. 22(b) shows the simple drawing of the target block, and Fig. 22(c) shows that two goliath cranes are synchronized and connected to the target block in sync.

The floating crane and both goliath cranes are modeled as multibodies based on multibody system dynamics. The wind force is also acting on all cranes and the target block. The floating crane is floating on the sea, so the hydrostatic and linearized hydrodynamic forces are acting on it. In addition, the mooring force is also acting on it.

5.3.2. Configuration of scenario

The mission is given as follows. The first job is to lift up a block. For this, the tug boats bring a barge, which carries the target block, alongside the dock gate. Then the operator moves a floating crane and two goliath cranes to the initial position. We assumed that these jobs are already done before starting the simulation. Thus, at the first of the simulation, all cranes start to lift up a block. After lifting up a block sufficiently, the barge will move away. Next, the operator starts to move the block inside the dock, and then they tried to lift down the block (see Fig. 23).

5.3.3. Simulation results

Fig. 24 shows the simulation results. Multiple wires connected the cranes and the blocks, and we chose some wires connected to the point ×6, ×7 and ×10, shown in Fig. 24(a). The graph in Fig. 24(b) shows the tensions acting on these wires. It shows that the tensions were changed according to the change of the following scenario: hoisting up a block, transporting, and hoisting down a block.

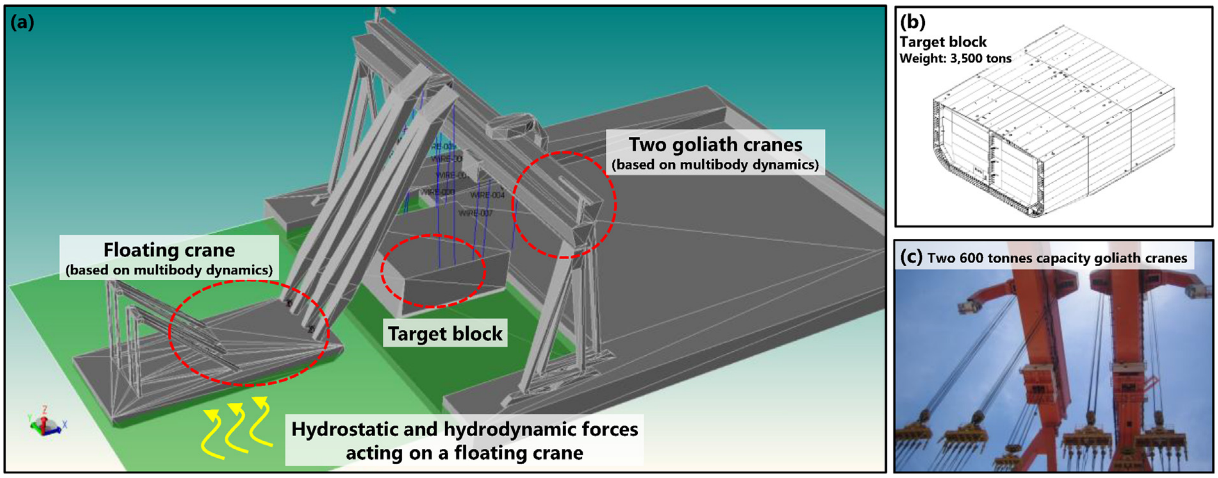

(a)
Two goliath cranes
(based on multibody dynamics)
Floating crane
(based on multibody dynamics)
Target block
Hydrostatic and hydrodynamic forces
acting on a floating crane
(b)
Target block
Weight: 3,500 tons
(c) Two 600 tonnes capacity goliath cranes

Fig. 22. Lifting and transporting a mega-block by using a floating crane and two goliath cranes: (a) system configuration, (b) a mega block for erection, and (c) two goliath cranes.

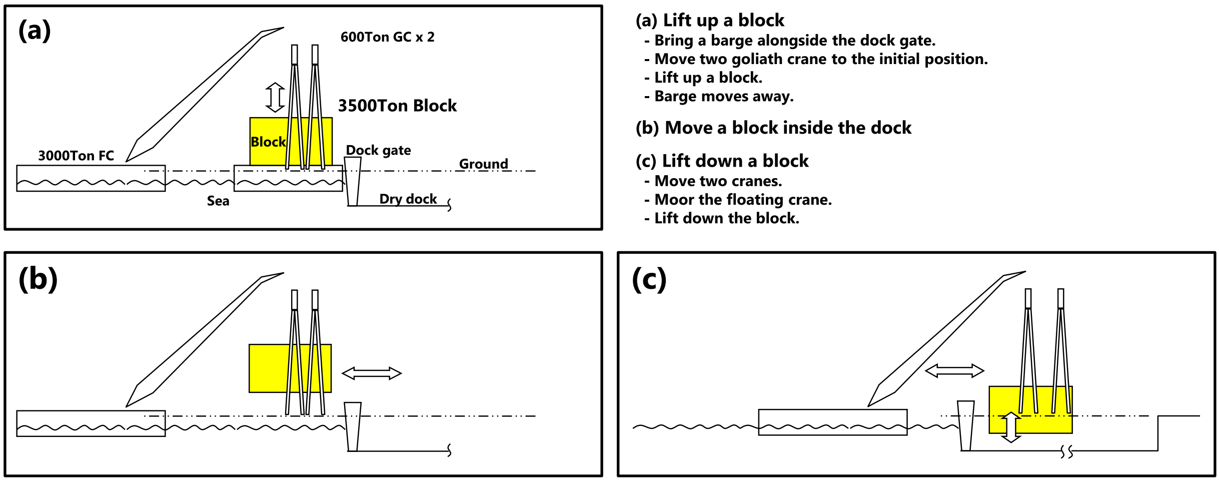


Fig. 23. Lifting and transporting a mega-block by using a floating crane and two goliath cranes: scenario configuration — (a) lifting up a block, (b) moving a block inside the dock, and (c) lifting down a block.

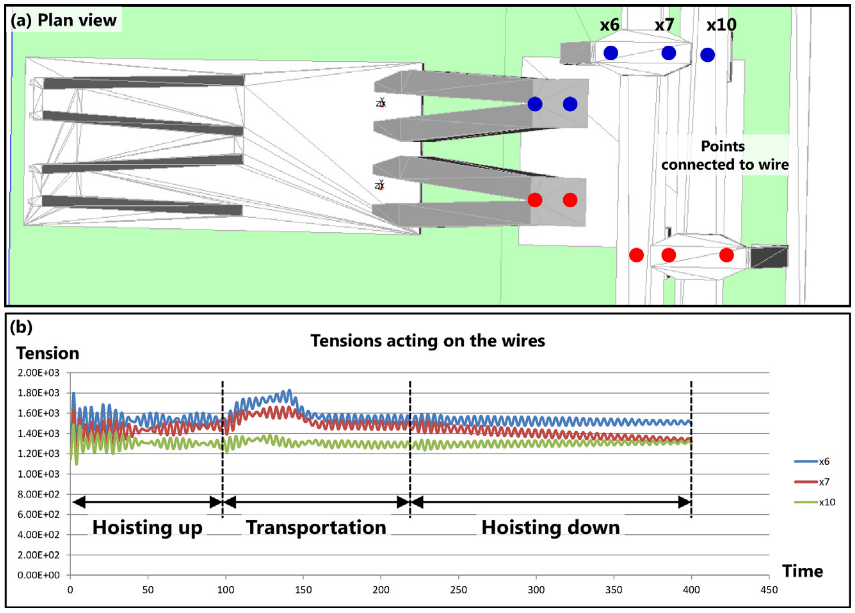


Fig. 24. Lifting and transporting a mega-block by using a floating crane and two goliath cranes: simulation results — change of tensions acting on the wires.

6. Conclusions

In this study, the scenario management method for multibody dynamics simulator was suggested to conveniently configure some partial operations of shipbuilding process such as lifting, turn-over, and erecting a block. The model design for the scenario management was proposed, and, in especial, a generalized actor model, which acts on the components of a multibody dynamics kernel, was proposed, considering its reusability and extensibility. All models for the scenario management was based on DEVS formalism, a formalism generally used on event-based simulation.

To evaluate the efficiency of the proposed scenario manager, it is applied to various examples of the simulation in shipyard. Three examples are introduced in this paper: double compound pendulum; block-lifting, transportation, and turn-over process; and lifting and transporting a mega-block. From these example, it was confirmed that the proposed method can applied to any examples in shipyard, and extend the functions for other exceptional situations and considerations.

In a future study, we will apply the developed program to various shipbuilding process simulations, such as to a simulation of the dynamic analysis of the offshore structure and the block assembly process, to improve the developed program's efficiency and applicability. In addition, the flexibility effect of the large steel structure such as a block and a floating crane will be also considered by adopting theories of flexible multibody system dynamics.

Acknowledgments

This work was partially supported by,

(a) the Human Resource Training Program for Regional Innovation and Creativity through the Ministry of Education and National Research Foundation of Korea, and

(b) the project "Development of Core Installation Technologies for Float-over and Dual Crane Methods", which is supported by the Ministry of Oceans and Fisheries.

References

Bang, K.W., 2006. Combined Discrete Event and Discrete Time Simulation Framework for Shipbuilding Process Planning (Master thesis). Seoul National University.

Cha, J.H., Ham, S.H., Lee, K.Y., Roh, M.I., 2010a. Application of a topological modelling approach of multi-body system dynamics to simulation of multi-floating cranes in shipyards, Proceedings of the Institution of Mechanical Engineers, Part K. J. Multi-body Dyn. 224 (4), 365−373.

Cha, J.H., Park, K.P., Lee, K.Y., 2010b. Numerical analysis for nonlinear static and dynamic responses of floating crane with elastic boom. Trans. Korean Soc. Mech. Eng. A 34 (4), 501−509.

Cha, J.H., Park, K.P., Lee, K.Y., 2012. Development of a simulation framework and applications to new production processes in shipyards. Computer-Aided Des. 44 (3), 241−252.

Cha, J.H., Roh, M.I., 2010. Combined discrete event and discrete time simulation framework and its application to the block erection process in shipbuilding. Adv. Eng. Softw. 41 (4), 656−665.

Cha, J.H., Roh, M.I., Lee, K.Y., 2010c. Dynamic response simulation of a heavy cargo suspended by a floating crane based on multibody system dynamics. Ocean. Eng. 37 (14−15), 1273−1291.

Cha, J.H., Roh, M.I., Lee, K.Y., 2010d. Integrated simulation framework for the process planning of ships and offshore structures. Robotics Computer-Integrated Manuf. 26 (5), 430−453.

Featherstone, R., 2008. Rigid Body Dynamics. Springer.

FunctionBay, Inc, 2003. RecurDyn^{TM} Solver Theoretical Manual.

Ha, S., Cha, J.H., Roh, M.I., Lee, K.Y., 2012a. Implementation of the submarine diving simulation in a distributed environment. Int. J. Nav. Archit. Ocean Eng. 4 (3), 211−227.

Ha, S., Ku, N.K., Roh, M.I., Lee, K.Y., 2012b. Cell-based evacuation simulation considering human behavior in a passenger ship. Ocean. Eng. 53, 138−152.

Hwang, I.H., Kim, Y.M., Lee, D.K., Shin, J.G., 2014. Automation of block assignment planning using a diagram-based scenario modeling method. Int. J. Nav. Archit. Ocean Eng. 6 (1), 162−174.

Ku, N.K., Ha, S., 2014. Dynamic response analysis of heavy load lifting operation in shipyard using multi-cranes. Ocean. Eng. 83, 63−75.

Ku, N.K., Ha, S., Roh, M.I., Lee, K.Y., 2012. Dynamic response analysis of multibody system in discrete event simulation. In: Proceedings of 1st International Conference on Operations Research and Enterprise Systems (ICORES 2012), Vilamoura, Algarve, Portugal, pp. 447−453.

Lee, D.K., Kim, Y.M., Hwang, I.H., Oh, D.K., Shin, J.G., 2014. Study on a process-centric modeling methodology for virtual manufacturing of ships and offshore structures in shipyards. Int. J. Adv. Manuf. Technol. 71 (1), 621−633.

Orlandea, N., Chace, M.A., Calahan, D.A., 1977. A sparsity-oriented approach to the dynamic analysis and design of mechanical systems − part 1 & 2. J. Eng. Ind. − Trans. ASME 99 (3), 773−779.

Schiehlen, W., 1990. Multibody Systems Handbook. Springer.

Shabana, A.A., 2005. Dynamics of Multibody Systems, third ed. Cambridge University Press.

Smith, R., 2006. Open Dynamics Engine v0.5 User Guide.

Ultramarine, Inc, 2013. An Introduction to MOSES.

Woo, J.H., Nam, J.H., Ko, K.H., 2014. Development of a simulation method for the subsea production system. J. Comput. Des. Eng. 1 (3), 173−186.

Zeigler, B.P., Praehofer, H., Kim, T.G., 2000. Theory of Modeling and Simulation. Academic Press, New York, NY.