Journal of Cleaner Production 281 (2021) 124443

Contents lists available at ScienceDirect

Journal of Cleaner Production

journal homepage: www.elsevier.com/locate/jclepro

# Efficient container virtualization-based digital twin simulation of smart industrial systems

Ting Yu Lin [a, b, c], Guoqiang Shi [a, b, c], Chen Yang [d, *], Yingxi Zhang [a, b, c], Jiezhang Wang [e], Zhengxuan Jia [a, b, c], Liqin Guo [a, b, c], Yingying Xiao [a, b, c], Zhiqiang Wei [a, b, c], Shulin Lan [f, **]

[a] State Key Laboratory of Complex Product Intelligent Manufacturing System Technology, Beijing Institute of Electronic System Engineering, Beijing, China
[b] Beijing Complex Product Advanced Manufacturing Engineering Research Center, Beijing Simulation Center, Beijing, China
[c] Science and Technology on Special System Simulation Laboratory, Beijing Simulation Center, Beijing, China
[d] School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
[e] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, Hubei, China
[f] School of Economics and Management, University of the Chinese Academy of Sciences, Beijing, China

ARTICLE INFO

*Article history:*
Received 22 April 2020
Received in revised form
25 August 2020
Accepted 28 September 2020
Available online 2 October 2020

**Handling editor**Cecilia Maria Villas Bôas de Almeida

*Keywords:*
Digital twin
Container virtualization
Efficient parallel simulation
Smart industrial service system
Cloud simulation
Simulation as a service

ABSTRACT

The great potential of digital twin (DT) in supporting smart industrial systems has brought huge requirements for on-demand DT-based simulation, a particularly useful and sustainable means, to assist various decision-making. However, there are major challenges to efficiently build and update the DT-based simulation system and provide simulation as a service (SimaaS): 1) virtualization machine based heavyweight methods to create simulation environments for DT models consume too much resource and time; 2) DT-based simulation systems in the cloud or developers' desktops could not well support the real-time response and synchronize with the physical counterparts at the edge of the network. Therefore, a methodology of container virtualization based simulation as a service (CVSimaaS) is put forward to utilize lightweight containers to realize convenient DT system deployment and less resource consumption with high efficiency. Then a device-edge-cloud system architecture with a formal process are proposed to support the CVSimaaS paradigm. A matrix based management and scheduling model for computing infrastructure, container images and services is established to support the efficient CVSimaaS process. Finally, the methodology is applied to building a DT-based simulation system for intelligent manufacturing. The results show that the DT-based simulation system can be 1) easily deployed to heterogeneous infrastructure and terminals at the cloud, edge and device, and 2) parallelly scheduled and operated on high performance cloud/edge on demand for large-scale online analysis.


## 1. Introduction

The new generation information and communications technology (ICT), such as the Internet of Things (IoT), Cyber-Physical System (CPS), Big Data and AI, promote the full connection and collaboration of human, products, industrial equipment, data, intelligent analysis and decision-making and execution systems, forming a new kind of the smart industrial system including human, physical space and cyberspace (Li et al., 2019). The smart industrial service system, a kind of Product-Service Systems (Zheng et al., 2018), is developed based on advanced ICT, aiming to provide smart services of data collection/storage, big data analytics, simulation and decision-making for empowering and optimizing industrial systems/applications. Stakeholders of industrial systems do not specifically demand physical products, per se, but rather are seeking the utility of these products and services. The smart industrial service system can deliver its performances and usage as a bundle to help improve the efficiency of industrial assets and processes, save materials/energy and avoid waste. In such a smart industrial service system, Digital Twin (DT), identified as one of the top 10 strategic technology trends by Gartner (Panetta, 2018), can play a key role in understanding, predicting, and optimizing the performance of industrial assets and processes in product/system design, configuration and operation stages (Digital. (2018). Digit, 2018; Leng et al., 2019; Leng et al., 2020).

* Corresponding author.
** Corresponding author.
*E-mail address:* yangchen666@bit.edu.cn (C. Yang).

https://doi.org/10.1016/j.jclepro.2020.124443
0959-6526/

According to the popular definition proposed by Glaessegen and Stargel, DT means an integrated multi-physics, multi-scale, probabilistic simulation of a complex product, which functions to mirror the life of its corresponding twin (Glaessgen and Stargel, 2012). DTs consist of components such as models, a set of analytic algorithms and knowledge. It is obvious from the above definition that simulation plays a vital role in DT based advanced analysis. It is a particularly useful and sustainable means when the industrial assets and processes costs too much to be built, physical experiments are dangerous or expensive, or it takes a long time to know the results of the system due to the changing parameters (Yang et al., 2018; Zeigler et al., 2000). Moreover, DT based simulation will act as one key part along with the physical systems. The higher the efficiency of simulation, the earlier the potential problems of the system could be predicted, so that measures could be taken in advance to avoid unnecessary losses and achieve sustainable development (Sonnemann et al., 2003; Lofgren and Tillman, 2011).

For such a DT-based simulation, a complex simulation system (e.g., comprising mechanical, electronic, fluid, control models) are first developed by engineers from different departments or enterprises and then distributed interactive simulation based on the models and corresponding simulation environments (solvers, platforms and OS (Operating System)) should be operated on demand (Li et al., 2017). Compared with traditional simulation (Leng et al., 2019; Zeigler et al., 2000; Li and Chai, 2002), there are several challenges along with the development of DTs.

(1) A DT is a dynamic virtual representation of its physical counterpart (system, process or service) and continually updated with real-time data of the latter physical object throughout its whole life cycle (Bolton et al., 2018). This means that simulation execution should not be conducted only on the desktop (of the model developer) which could not satisfy the requirement of real-time response and adjustment. There exists unbearable communication latency between the cyber model and the physical system (Leng et al., 2019), which is largely far away at the network edge. Therefore, we propose to migrate and deploy the DT-based simulation system to the edge near the physical counterpart, to better synchronize the DT models and the physical objects. However, it is challenging to implement the migration and deployment, due to the heterogeneity of edge infrastructure and terminals.
(2) A DT could be developed using artificial intelligent and adjusted by real-time data, and could be executed to help understanding, learning and reasoning of its corresponding twin (physical counterpart) (Bolton et al., 2018). This means that the DT-based simulation system should not only be used for traditional small-scale off-line verification which could not support dynamic deep analysis (Leng et al., 2020), but also be scheduled and executed in parallel on a high performance cloud or edge/end nodes on demand where large-scale online simulation is executed for model training and scheme search. However, it is challenging to efficiently provide, schedule and execute multiple instances of the complex DT-based simulation system in the cloud or edge nodes to accelerate the simulation and get optimal solutions.

Considering the requirement of building and executing a DT-based simulation system on different computing devices (cloud, computer, mobile terminal, etc.) on demand, the Simulation as a Service (SimaaS) (Li et al., 2009a; Tsai et al., 2011) which could support convenient and continuous system deployment on multi-terminals as well as automatic and efficient scheduling of multiple system instances becomes more and more important. A key technology to realize the SimaaS for different users is virtualization technology (Zhang et al., 2011; Ren et al., 2012), which could provide a unified encapsulation of the simulation system as a virtual machine (VM) image for heterogeneous devices, support the mode of one-time image development and repeated rapid deployment, and satisfy the demand for dynamic construction and efficient execution of a complex DT-based simulation system. A VM image contains the DT-based simulation system and the runtime environment (solvers, platforms and OS) for running the simulation. After a VM image is transmitted and deployed to a host computing device with a hypervisor (or VM monitor), the DT-based simulation system can then be constructed by creating and running a VM on the device from the VM image.

However, the traditional hypervisor virtualization still has shortcomings: the size of VM images is very large in Gigabytes and VMs may take up a lot of resources during their transmission, deployment and execution on heterogeneous devices. Thus the VM based method is not so efficient and fast and could not well adapt to large-scale online analysis on demand (Li et al., 2018). Moreover, computing resources on edge nodes or end devices are usually limited to support large-scale VM based simulation. The new generation of virtualization - container (Dua et al., 2014; BernsteinDavid, 2014) provides a promising solution. Instead of virtualizing the hardware like a VM, a container virtualizes at the OS level. Containers run at a layer on top of the host OS and they share the OS kernel. Containers have much lower overhead than VMs and a much smaller footprint. Therefore, a lightweight container has the advantages of convenient and consistent deployment, less resource consumption and high efficiency. There are research and practices about container-based industrial applications (Rufino et al., 2017). However, there still exist major issues (paradigm, architecture, process, resource scheduling models) to be explored in the development, deployment and execution process of a complex DT-based simulation system as a new kind of unique container based industrial applications. Therefore, to support on-demand simulation of smart industrial system in parallel on a high performance cloud or an edge node, this paper proposes a novel methodology towards Container Virtualization-based Simulation as a Service (CVSimaaS) supporting parallel DT based advanced analysis.

The rest of the paper is organized as follows: Section 2 introduces related work; Section 3 proposes a new paradigm supporting DT based advanced analysis; Section 4 discusses approaches to support the new CVSimaaS paradigm; Section 5 presents a management and scheduling model for CVSimaaS; Section 6 presents an application example of our methodology in intelligent manufacturing workshop scheduling; Section 7 concludes the paper with our contributions and future work.

## 2. Related work

### 2.1. Digital twin-based simulation

A DT is a digital set model of physical objects (Yang et al., 2018; Zheng et al., 2018). The DT-based simulation could be used to analyze, predict, diagnose, train, etc., the results of which could be fed back to physical objects, so as to help optimizing and decisions making for physical object (Shen et al., 2020). Simulation merges the physical and virtual world in all life cycle phases, e.g. to support design tasks or to validate system properties, to optimized operations and predict failure during operation and for service (Boschert and Rosen, 2016). The review shows that DT-based simulation has been widely used in industries and other fields (Trung, 2020; Blasco et al., 2020), e.g., for joint optimization of operations in the large-scale automated warehouse product-service system (Leng et al.,

2019) and for supporting rapid reconfiguration of the automated manufacturing system via an open architecture model (Leng et al., 2020). However, it also brings new problems that traditional simulation does not meet. On the one hand, DT-based simulation should be integrated with pervasive computing (Karakra et al., 2018), and should overcome challenge about keeping model components consistent (Derler et al., 2012). On the other hand, DT-based simulation should be operated faster than real time (Zhidchenko et al., 2018), and should be operated in parallel supporting management and control (Wang and Wang, 2010). Overall, DT-based simulation should support consistent deploying on multi-terminals and automatic scheduling of multi-instances, and SimaaS becomes very important.

### 2.2. Simulation as a service

As the scale of simulation systems becomes larger and the scope of simulation stakeholders becomes wider, the simulation gradually develops to adopt service-oriented approaches. An Extensible Modeling and Simulation Framework (XMSF) was proposed with the principles to link models by Web services (Buss and Ruck, 2004). Simulation Grid then promoted the wide sharing and collaboration of live, virtual and constructive simulation resources (Li et al., 2005). Based on the virtualization of simulation resources, cloud simulation improved the multi-users sharing capability of simulation resources in simulation grid, and supports efficient fault-tolerant migration of simulation tasks (Li et al., 2009b). With the rapid development of cloud computing, SimaaS has developed rapidly. There were researches on the combination of simulation high level architecture and cloud computing multi-tenants architecture to develop a new simulation framework (Dragoicea et al., 2012). Some research transformed the parallel simulation system to make full use of the cloud computing environment for efficient operation (Malik et al., 2009). Our team earlier proposed consistent deploying on distribute infrastructures by managing the multi-centric resource and capability (Lin et al., 2015), and proposed automatic scheduling of multi-instances to cloud simulation based environment for optimization (Guo et al., 2017). All these depend on the application of virtualization technology in simulation which should be adapted to simulation.

### 2.3. Virtualization technology in simulation

Virtualization technology has been proposed for supporting concurrent to a mainframe computer by IBM since 1960 (Anderson et al., 2005), and has become a key technology in cloud computing (Buyya et al., 2009). A Toroidal Large Hadron Collider Apparatus (ATLAS) simulation experiment introduced virtualization technology to enhance security of simulation systems and increase utilization of CPU (Sailer et al., 2005). The botnet evaluation environment (BEE) introduced virtualization technology to encapsulate various kinds and versions of operating systems for automatic establishment of simulation environment (Barford and Blodgett, 2007). The Virtualization-based simulation platform (VSIM) introduced virtualization technology to support multidisciplinary models' collaboration for complex product design (Ren et al., 2012). The traditional virtualization technology based on hypervisor has the problems of long start-up time and high infrastructure occupancy. Compared with Xen (Barham et al., 2003) and other traditional virtualization projects or products, the new generation of virtualization technology such as container has more advantages (Khalid et al., 2017). The container technology represented by docker mainly encapsulates the process and realizes the isolation of the process running environment in each container by extension of LxC method (BernsteinDavid, 2014). Container has the advantages of convenient and consistent deployment, less resource consumption and high efficiency, but it also has some disadvantages such as weak isolation, which has become a research hotpot in the field of virtualization (Song et al., 2019). Li et al. introduced virtualization technology into SimaaS field earlier (Li et al., 2011), and then proposed smart cloud simulation which earlier introduced container as the approach of virtualization (Li et al., 2018). Taking the advantages of container, and overcoming the shortcomings of container, the application of container technology as the virtualization approach in the field of simulation is a relatively innovative work today shown as follow: (1) Container oriented unified abstraction for the DT models, their solvers and dependent platforms together with the infrastructure should be made to adapt to the complexity of the DT-based simulation system supporting construction and operation. (2) Container oriented management and scheduling for the DT models, their solvers and dependent platforms together with the infrastructure should be built to address the seamless use of cloud, edge and end such as simulation for model training and scheme search.

## 3. CVSimaaS: a new paradigm supporting DT-based simulation

### 3.1. DT-based simulation on cloud, edge and end

A DT is a digital set model of physical objects including the digital simulation model, the cognitive model and the decision-making model. The design and operation of a DT shown in Fig. 1 relate to the whole CPS on the cloud, edge and end.

In the design stage, the models especially the physical models involved disciplines of mechanical, electronic, fluid, control and so on would be designed by different enterprises or departments on different edges commonly. Then, the distributed interactive simulation system based on corresponding different solvers and platforms would be integrated and constructed collaboratively. The simulation cloud as a shared environment provides the best way for integration and construction, and it is also convenient to provide services for subsequent operation (Yang et al., 2012).

In the operation stage, the cognitive model and the decision-making model would be loaded into the physical object which is an embedded CPS system. When the system deals with the external environment, the cognitive model and decision making model play the core role in the loop of Observation-Orientation-Decision-Action (OODA) (BryantDavid, 2006). The physical model corresponded to and mapped the physical object and its environment could predict the action effect of the physical object, hence scheme search and verification would be carried out on the simulation edge through the parallel DT-based simulation. In order to deal with the uncertainty in reality, DT based cognition and decision-making would be evolutionary through large scale parallel simulation and reinforcement learning of DTs in the simulation cloud (Lin et al., 2020). The new DT model would be migrated to the simulation edge which would be carried out adaptive training combined with more actual data. Finally, the new DT model would be updated to the embedded CPS system.

### 3.2. Container virtualization-based simulation system

To better illustrate the new paradigm supporting DT-based simulation, it is important to first discuss the container-based simulation system compare with the traditional VM based simulation system.

Traditional virtualization would encapsulate the entire OS, application and its dependent libraries into a VM which usually occupies tens of GB storage of physical machine (PM) and takes

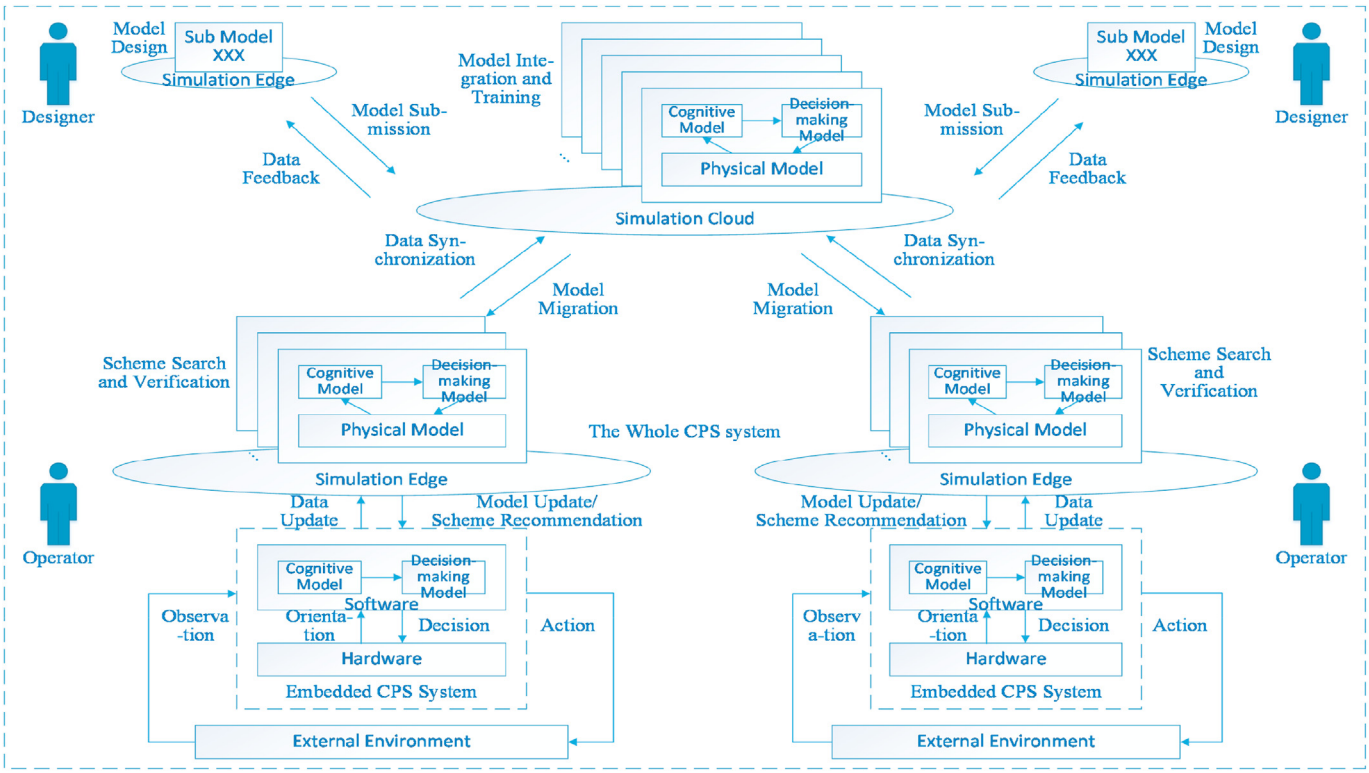


**Fig. 1.** DT-based simulation on the cloud, edge and end.

several minutes to start up. A traditional virtualization based simulation system is shown in Fig. 2. It significantly improves the transplantation capability of the DT model with its solver and dependent platform, and promote the concurrent shared utilization of infrastructure for simulation. However, because the VM has the problems of high infrastructure occupancy and long start-up time, it is difficult to transmit the VM image between cloud and edge for model migration and update, and it is not high-efficiency to instantiate VMs on the terminals of cloud, edge and end for timely simulation, especially if it is a complex distributed interactive simulation system. In addition, we have to encapsulate the DT model with its solver and dependent platform into one VM image which would make the granularity of simulation services too big and the flexibility of sharing insufficient.

A new generation of virtualization technology would only encapsulate the application and its dependent libraries into the container which usually occupies tens of MB storage of PM and takes several seconds to start up. The containers would run as the independent processes which share the underlying OS kernel through the container engine. At the same time, containers provide services to each other through interfaces, so containers could be composited and integrated very flexibly. A container-based simulation system is shown in Fig. 3. The DT models with their solver and dependent platform are encapsulated in container image and been run in process separately. They would be composited on demand to construct a DT-based simulation system, even if it is a complex distributed interactive simulation system. As the size of a container image is far smaller than a VM image and the start-up speed of a container is much faster than that of a VM, the size of the simulation system is small enough to transmit and deploy between cloud and edge easily, and the scheduling of the whole simulation system is much more efficient. In addition, it means more fine-grained simulation resource encapsulation which could enable flexible sharing, supporting incremental image assembly and deployment.

### 3.3. CVSimaaS supporting DT-based simulation

CVSimaaS is a new SimaaS paradigm supporting DT-based simulation on cloud, edge and end. Based on the new generation virtualization technology of container, the infrastructure of cloud, edge and end would be virtualized, and the DT models with their solvers and dependent platforms would be encapsulated into the container image. The infrastructure and the container image would be registered, managed and operated as the service in a logical service pool. Users could search, match, composite and access the services through the portal and the network on demand anytime and anywhere. Then the container images of the DT models with their solvers and dependent platforms would be instantiated as the containers on optimized terminals of cloud, edge or end, the DT-based simulation system instance would be efficiently constructed and operated to complete all kinds of activities in the whole life cycle.

The paradigm has two features shown as follow effectively utilized the advantages of container.

(1) With the advantages of convenient and consistent deployment, the container could be easily scheduled to different cloud, edge and end which means that DT-based simulation could be constructed and operated on demand anytime, anywhere. In addition, the container could be easily instantiated on the terminals of cloud, edge and end which means that one-time design, multiple time deployments and operation without filed debugging.
(2) With the advantages of less resource consumption and high efficiency, the DT-based simulation system instance with the

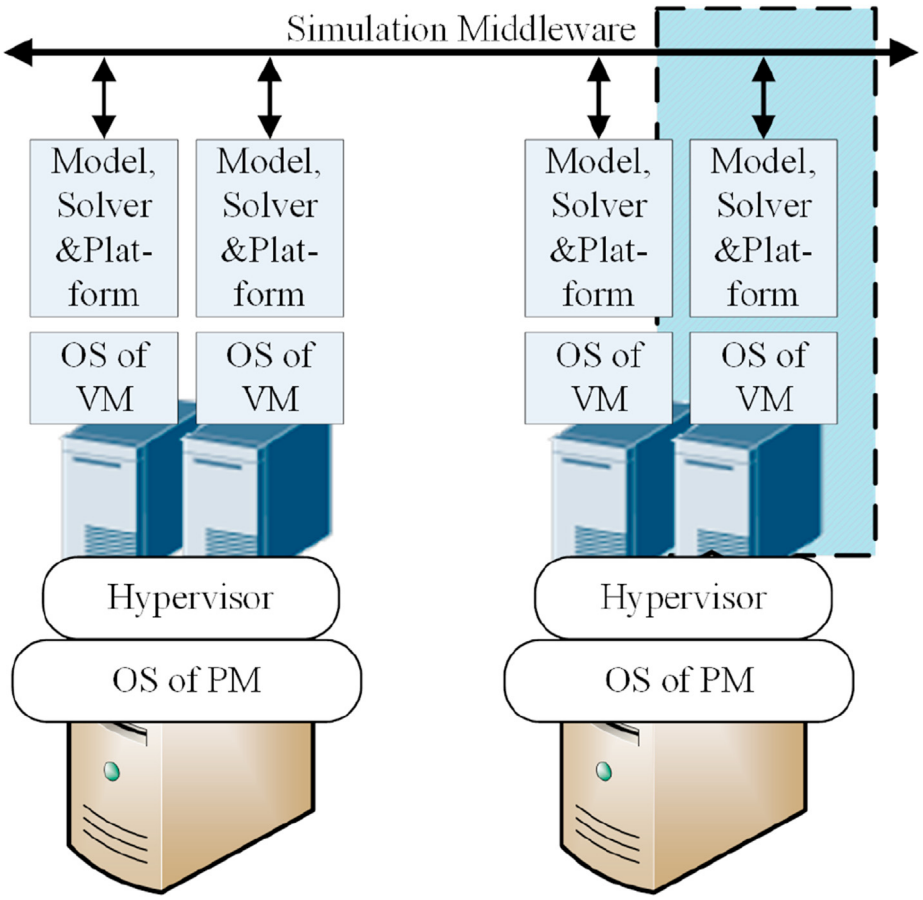


**Fig. 2.** Traditional VM based simulation system.

distributed interactive characteristic commonly could be easily integrated and constructed on one terminal which could greatly improves the efficiency of debug and operation. In addition, the DT-based simulation system instances could be efficiently constructed many times on demand for large-scale model training on the cloud and parallel scheme search on the edge.

When the new paradigm brings benefits, it also causes new problems to be solved. As DT-based simulation must be synchronized with the physical system (Yang et al., 2018; Bennett et al., 2015), the appropriate cloud, edge nodes or end devices needs to be selected optimally to deploy and operate the containers. One big challenge is to generate simulation results no later than the stringently required time for the physical objects/systems (Yang et al., 2018). In addition, the management and collaboration of cloud, edge, and end resources become more and more urgent to meet the stringent requirements of latency, security and privacy sensitive, or geo-distributed industrial applications (Yang et al., 2020).

To address these issues, the research adopts the methodology shown in Fig. 4. Firstly, the architecture and the process of CVSimaaS should be proposed to manage and schedule the infrastructure of cloud, edge, and end which could also enable deploy and operate the container. Secondly, a descriptive model should be put forward to formalize the infrastructure, container image and service, then an analytical model could be adopted for optimization of the management and scheduling.

## 4. Approach to support the new paradigm CVSimaaS

### 4.1. Architecture of CVSimaaS

According to the architecture of cloud simulation (Li et al., 2011), the architecture of CVSimaaS shown in Fig. 5 adopts container as the new virtualization approach, and newly developed an edge processing platform layer, to full support the DT-based simulation. The layers are explained as follows:

(1) Physical layer

Physical layer mainly consists of simulation resource/product/capability corresponding to the physical space of the whole CPS system in Fig. 1. The simulation resource includes the whole infrastructure of cloud, edge and end together with DT models and their solvers and dependent platforms which belong to the infrastructure. The simulation product refers to the embedded CPS system of the real product or its emulator. The simulation capability usually refers to the human who designs or operates the DT.

(2) Access/communication layer

Access/communication layer consists of the interface, device and network using to connect and integrate the simulation resource/product/capability of cloud, edge and end. It contains the perception and control interface of the simulation resource/

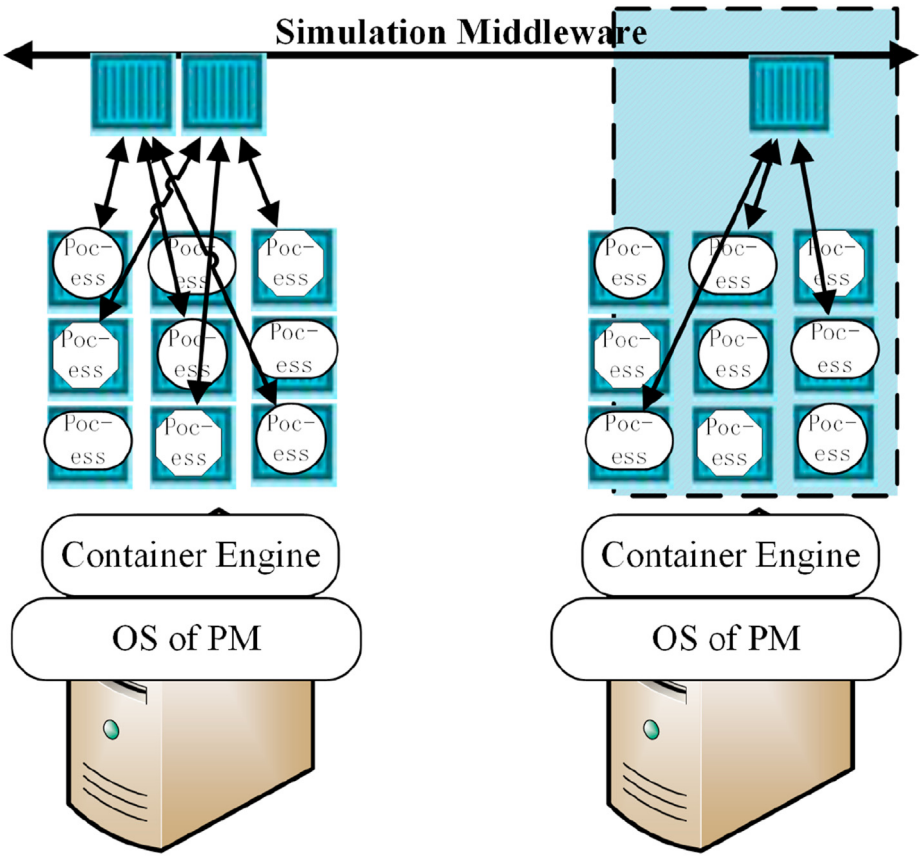


**Fig. 3.** Container-based simulation system.

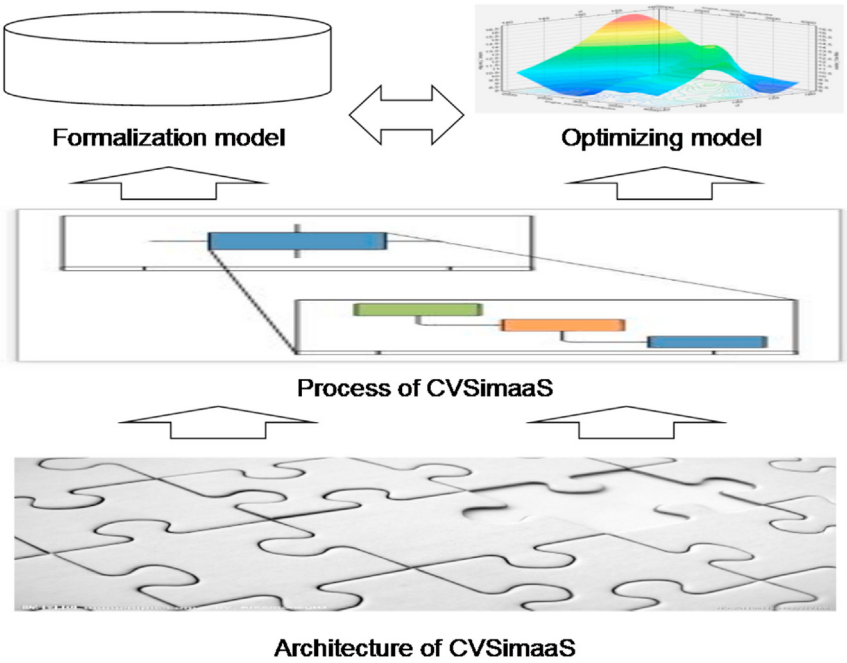


**Fig. 4.** The research methodology of this paper.

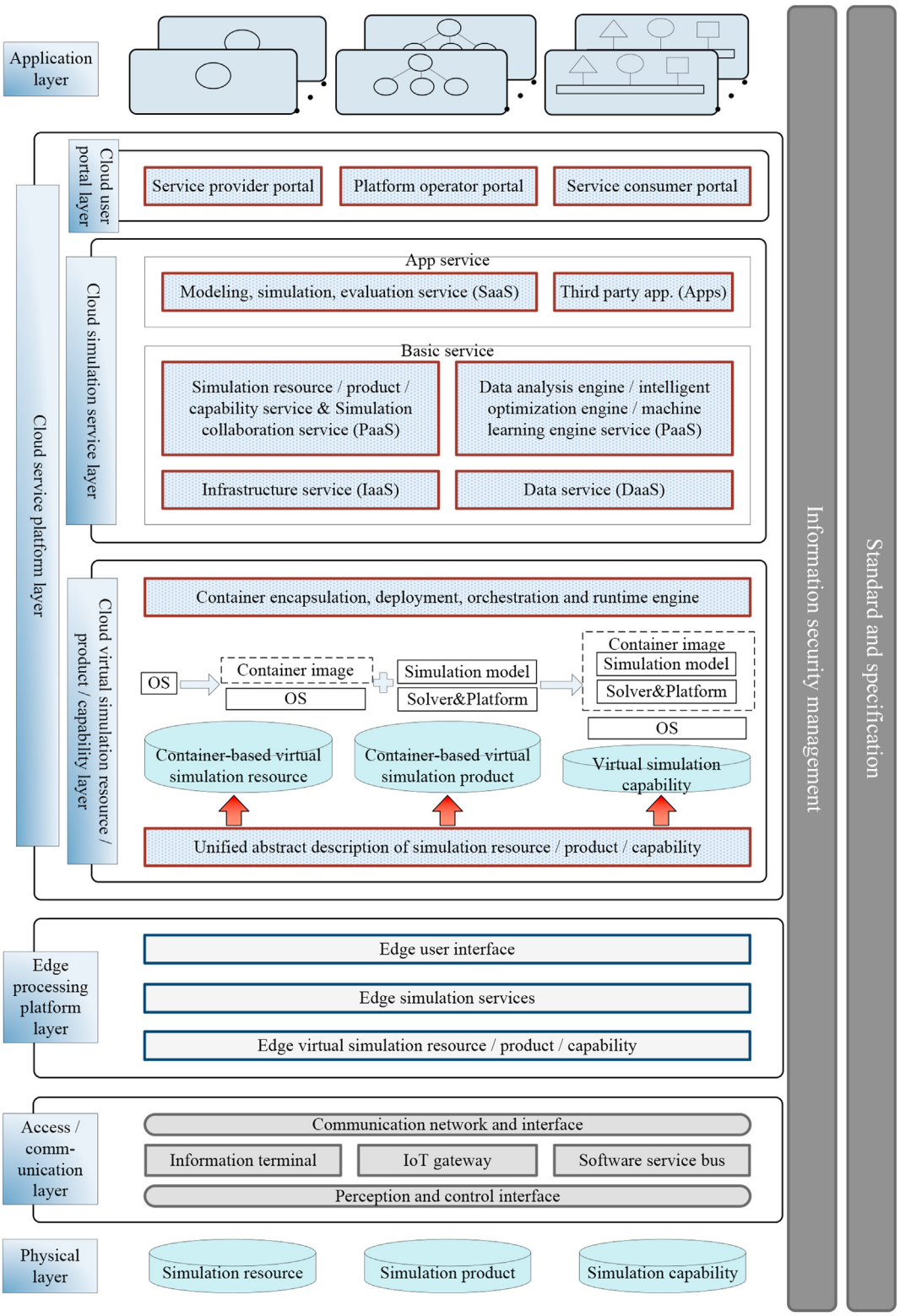


**Fig. 5.** Architecture of CVSimaaS.

product/capability which could be accessed by information terminal, IoT gateway, software service bus. It also contains the communication network and interface of cloud, edge and end which could realize the interconnection with the cyberspace of the whole CPS system.

(3) Edge processing platform layer

Edge processing platform layer consists of edge virtual simulation resource/product/capability, edge simulation services and edge user interface. Edge processing platform layer is similar to its

upper layer cloud service platform layer, but it is characterized by less infrastructure scale, more real-time service, user interface adapted to pervasive interactive terminal.

(4) Cloud service platform layer

1) Cloud virtualization layer

Cloud virtualization layer consists of container-based cloud virtual simulation resource, container-based cloud virtual simulation product and cloud virtual simulation capability. All of them are abstracted unified, and the simulation resource and product are adopted container-based virtualization driven by container encapsulation, deployment, orchestration and run-time engine.

2) Cloud simulation service layer

Cloud simulation service layer consists of basic services and app services. The basic service includes the infrastructure service known as IaaS (Manvi and Krishna Shyam, 2014), the data service known as DaaS (Wang et al., 2017) and the core platform service known as PaaS (Lawton, 2008) which contains the simulation related platform service and the analysis related platform service. The simulation related platform service mainly supports registered, publication, discovery, search, matching, composition and collaboration of virtual simulation resource/product/capability. The app service includes multi-tenant services known as SaaS (Benlian and Hess, 2012) mainly provided by platform (to support the life cycle of a DT such as modeling, integration, simulation, evaluation) mainly, and other third party application services complementally.

3) Cloud user portal layer

Cloud user portal layer consists of service provider portal (i.e. the designer of a DT), service consumer portal (i.e. the operator of a DT) and platform operator portal for different roles in the whole CPS system.

(5) Application layer

Application layer consists of all types of complex DT-based simulation systems followed the specifications such as FMI (Fritzson, 2014) or HLA (Lasnier et al., 2013), supporting all kinds of activities in the whole life cycle based on CVSimaaS.

### 4.2. Process of CVSimaaS

Based on the various kinds of services in the architecture, the new paradigm CVSimaaS is realized in the following process. According to the specification IDEF0 (Marca and Mcgowan, 1993), the main process of CVSimaaS and the sub processes with DT design and operation are discussed (Figs. 6–8).

(1) The main process of CVSimaaS

Based on the cloud/edge processing platform, A0 is activated by the request of service provider portal, and then design the DT which would be encapsulated in container images as the output. The output would be passed to A1. Also based on the cloud/edge processing platform, A1 is activated by the request of service consumer portal, and then operate the DT to generate the simulation result. If there are no appropriate container images, the requirement would be reported back to A0.

(2) Sub-process of DT design

The sub process is activated by the request of service provider portal. Based on the modeling service, A00 develops the model with the requirement for design of a DT. New model with its solver and dependent platform would be passed to A01 where both of them would be encapsulated in the container image based on the container engine. Then, the container image would be stored in A02 based on the infrastructure service. For complex DT-based simulation which comprises of several models, the models would be integrated in A03 based on the integration service. Both the A01 and A03 would generate the description of the container image. The information of the container image's storage address and description would be passed to A04 where registration, publication and discovery would be done based on the simulation resource service. After that, the container images which are the output of the sub process could be used for the DT-based simulation.

(3) Sub process of operation of a DT

The sub-process is activated by the request of service consumer portal. Based on the simulation service, A10 analyzes the simulation task with the requirement for operation of DTs. Service description would be passed to A11 where search, matching,

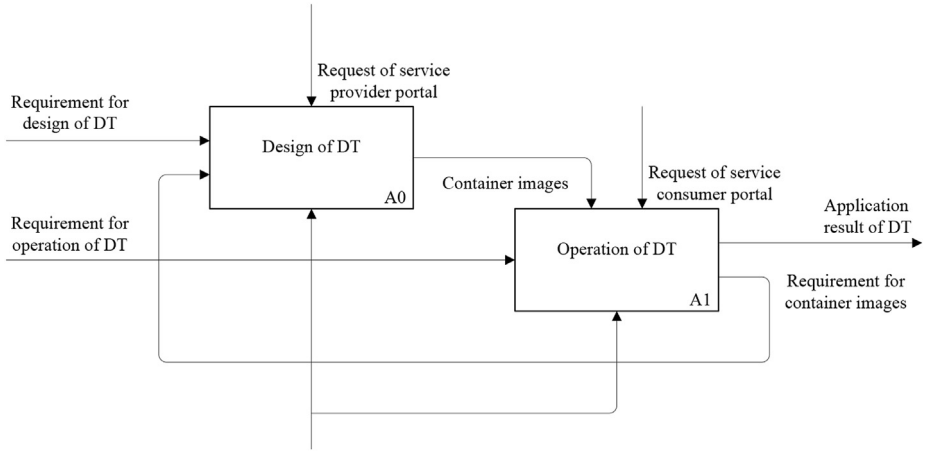


**Fig. 6.** Main process of CVSimaaS.

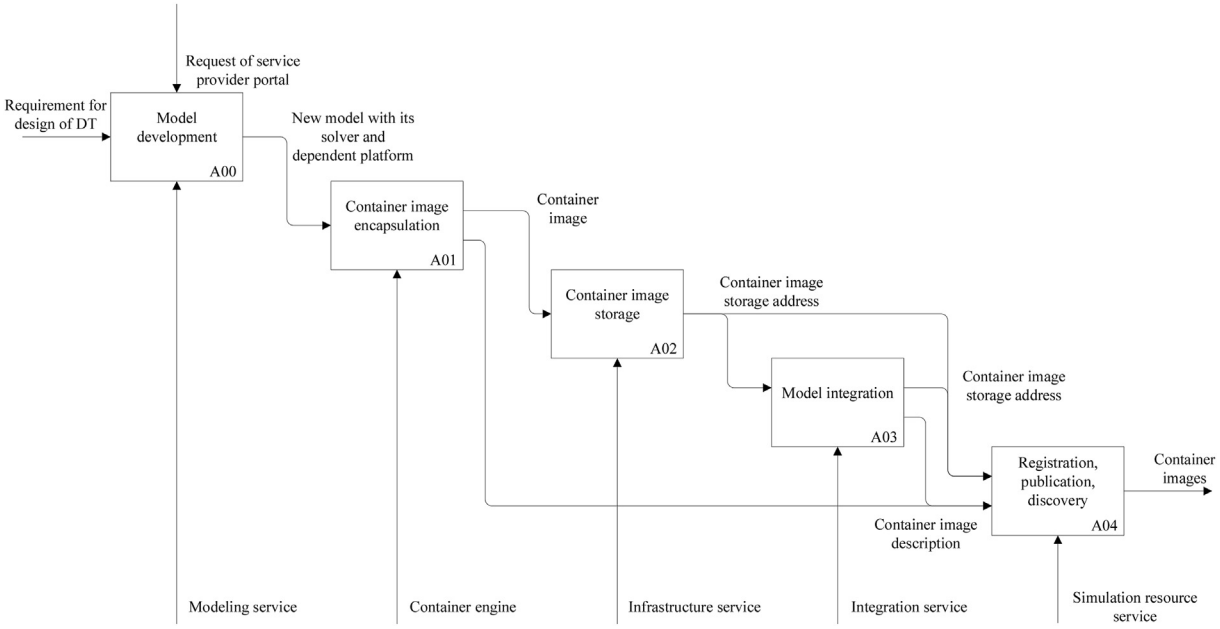


**Fig. 7.** Sub-process of DT design.

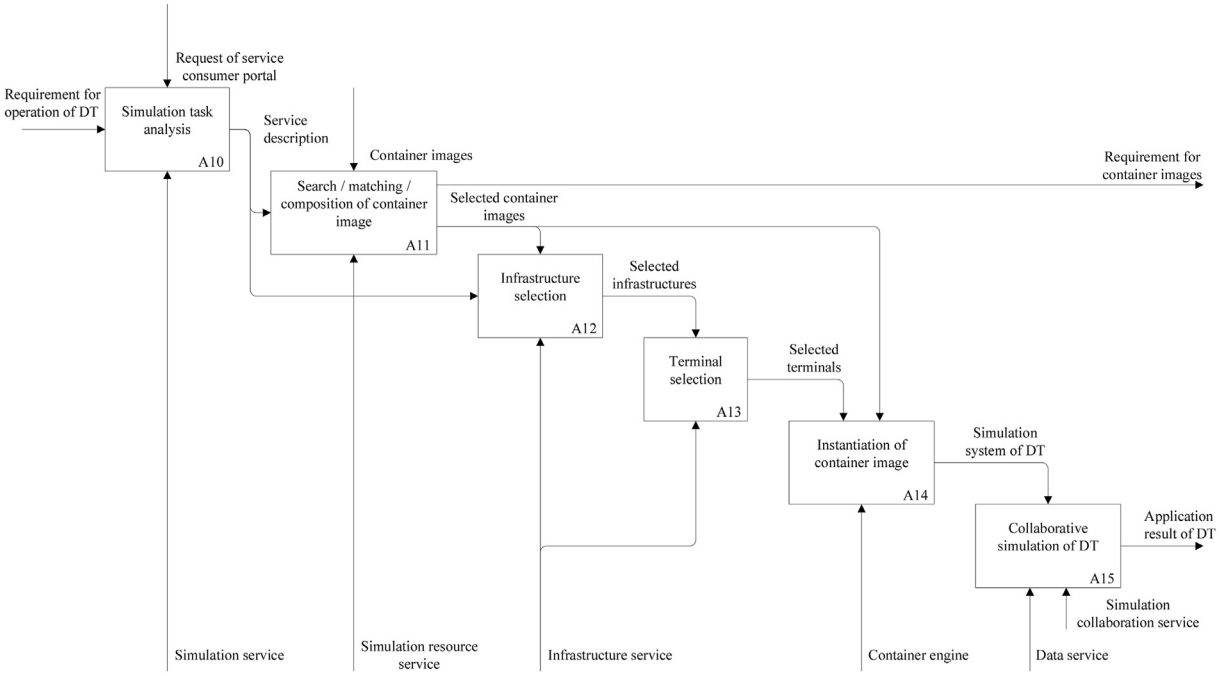


**Fig. 8.** Sub-process of operation of a DT.

composition of container images would be done based on the simulation resource service. If there are no appropriate container images, the sub process would finish and output the requirement for container images. Else, selected container images would be an activated condition of A12 in which the decision for which infrastructure to operate the DT-based simulation would be made based on the infrastructure service according to service description. And then, selected infrastructure would be an activated condition of A13 in which the decision for which terminal to run the DT-based simulation would be made also based on the infrastructure service. When both the container images and terminals are selected, A14 would instantiate the container based on the container engine to generate the DT-based simulation system. After that, collaborative DT-based simulation would be operated to generate the simulation result based on the data service and the simulation collaboration service.

## 5. Model of management and scheduling for CVSimaaS

### 5.1. Formalization model for management and scheduling

To better support management and scheduling in the above process of CVSimaaS, especially search/matching/composition of container image, infrastructure/terminal selection, a formal model is put forward to describe the infrastructure, container image (basic image, incremental image and composite image) and service

(requirement and supply). All levels of descriptions follow the multi-view (including static attribute view, service model view and operation management view) description model proposed by the authors' team in (Lin et al., 2019). The detailed descriptions include Definition 1 to Definition 6.

(1) Definition 1 Static Attribute

StaticAttri = < Id, Name, Dscrp, Type, Func< {Para} >,

ResGroup, Provider, Form, Dpndnt< {Para'} >, Others > (1)

Definition 1 shown in (1) denotes the static attributes such as id, name, description, type, function, resource group, provider, service form, dependency, in which the function should be represented by key parameters and would be referenced by the dependency of other resource.

(2) Definition 2 Atom Service Model

AtomSvrModel = < Ports < Ports_In, Ports_Out >, States, Response

< #States, Behavior, Trigger, Target< Duration >, ExitAction > >(2)

Definition 2 shown in (2) denotes the atom service model such as ports, states, response of state transition using finite state machine, which could illustrate the behavior requirement and supply of the service supporting search/matching/composition.

(3) Definition 3 Coupled Service Model

CoupledSvrModel = < This, SubSvrModel, Ports < Ports_In, Ports_Out >,

Coupling, Scheduling< #SubSvrModel, InfledSvModel > > (3)

Definition 3 shown in (3) denotes the coupled service model such as subordinate service model, ports and their coupling, scheduling of sequence using sequence diagram, which could illustrate the behavior requirement and supply of the service supporting search/matching/composition.

(4) Definition 4 Delivery of Service

Grounding = < Form< #Ports, Envvar, Prtcol, Mechnm > > (4)

Definition 4 shown in (4) denotes the service delivery approach including ports in service model, environment variable to be set, access protocol and calling mechanism for each service form which should be agreed between provider and consumer.

(5) Definition 5 Quality of Service

QoS = < SLA< {Factor}, Cost >, {Satsf}, Others > (5)

Definition 5 shown in (5) denotes the quality of service (QoS) including objective indicators described by service level agreement and subjective indicators evaluated by consumer, which are important basis for optimization supporting resource selection.

(6) Definition 6 Dynamic Attribute

DynamicAttri = < PhyRes, {Utiliz}, LifeCycle, Alloc, Resrv > (6)

Definition 6 shown in (6) denotes the dynamic attribute including associated physical resource set, resource utilization, state of life cycle, allocated and reserved status. LifeCycle is a set {IDLE, CONFIGURED, RESERVED, OCCUPIED, UNAVAILABLE, ERR, DESTROYED}.

There are some focuses in the multi-view description model when it is used in management and scheduling for CVSimaaS. The characteristics of containers should be considered to support construction and operation of a DT-based simulation system and help overcome the shortcoming of container in the process. The details are shown in Fig. 9.

(1) Infrastructure description

The infrastructure's function could be defined by key parameters including CPU number, memory size and network speed. And the QoS of infrastructure could be defined by available CPU number, memory size and network speed. Its associated physical resource set, resource utilization, state of life cycle should be assigned value to support infrastructure/terminal selection.

(2) Basic image description

The basic image's function could be defined by the OS, and its dependency is the infrastructure with the specific ID or key parameters including CPU number, memory size and network speed. The service form of a basic image could be defined by ports, environment variable and protocol. And the QoS of a basic image could be defined by indicators of image size and starting time. For the incremental image and composite image must depend on the basic image, the state of life cycle could only be assigned value at this level.

(3) Incremental image description

The incremental image's function could be defined by the solver and platform, and its dependency is the basic image with the specific ID and the infrastructure with key parameters including CPU number, memory size and network speed. The behavior of an incremental image about the model encapsulated in it could be described by atom service model. The service form of an incremental image could also be defined by ports, environment variable and protocol, where the communication port should be set in environment variable that could help avoid conflict when implement large-scale online simulation. The QoS of an incremental image are the same and associated with the QoS of a basic image.

(4) Composite image description

The composite image's function could be defined by the model, and its dependency is the incremental image with the specific IDs and the infrastructure with key parameters including CPU number, memory size and network speed. The behavior of a composite image about the model encapsulated in it could be described by coupled service model, and it is associated with the atom service model of an incremental image. The service form of a composite image is the same with the incremental image. The QoS of a composite image contains the same indicators with the QoS of an incremental image associated, and it could be defined by more indicators such as the communication efficiency with the physical object.

(5) Service description

The instance of the container image which runs on the selected infrastructure/terminal provides simulation services to users. The

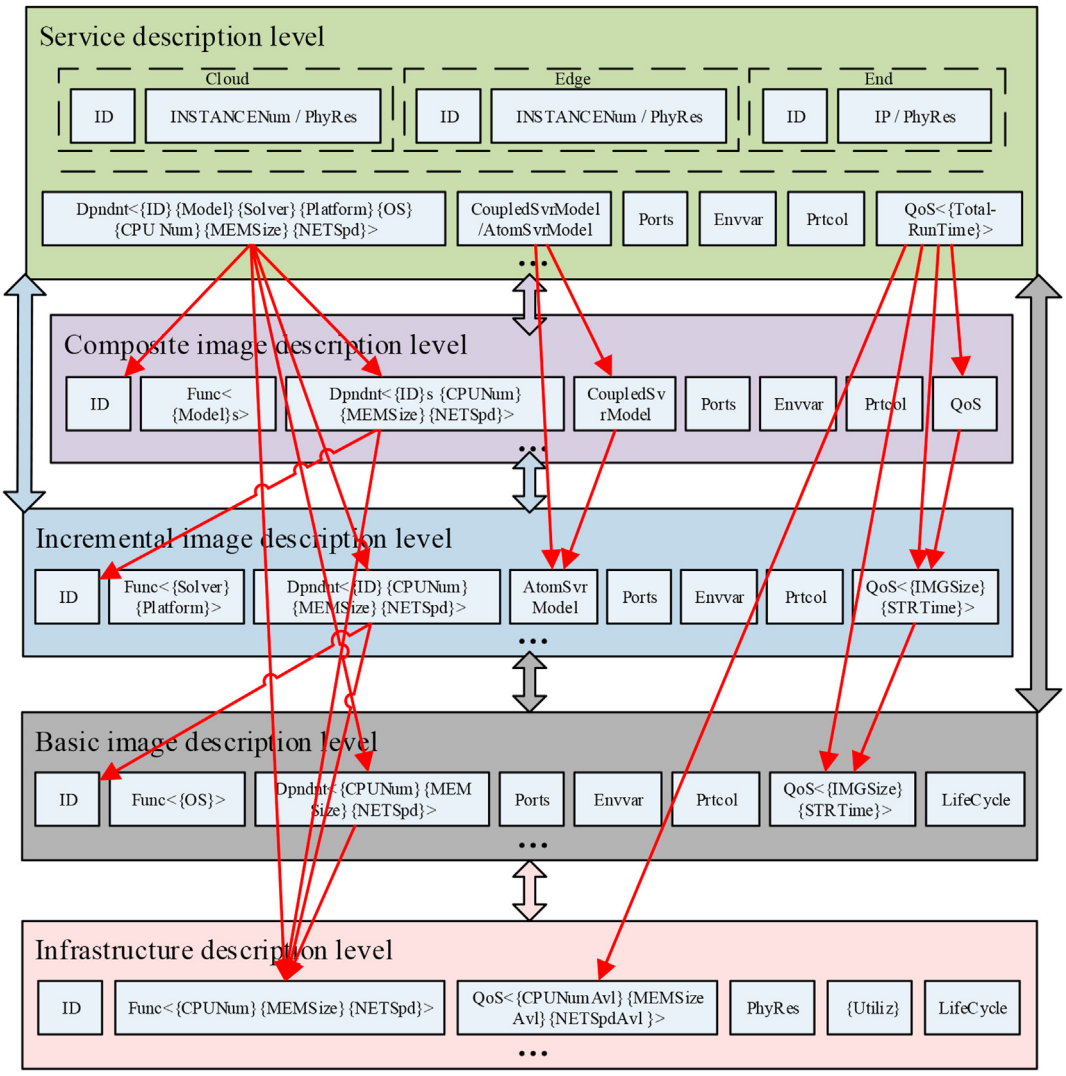


**Fig. 9.** Formalization model for management and scheduling.

service depends on the specific ID of the image or the DT model with its solver and dependent platform encapsulated in the image, and it also depends on the infrastructure with key parameters including CPU number, memory size and network speed. The behavior of service could be described by atom service model or coupled service model, and it is associated with the atom service model of an incremental image or the coupled service model of a composite image. The above two could be used to support search/matching/composition of a container image. Next, the QoS could be defined by the indicator of total run time, and it is associated with the QoS of infrastructure, basic image, incremental image and composite image which is useful for optimization supporting resource selection discussed later. In addition, the service form of service defined by ports, environment variable and protocol should be chosen and configured. Provider should describe the associated physical resource set where the service is alive, and consumer should describe whether cloud, edge and end would be used to support the construction of a DT-based simulation system and how many instances would be operated.

### 5.2. Optimizing model of management and scheduling

In the whole CPS system, multiple duplicates of the DT-based simulation system, according to the requirement, could be scheduled, initialized with different parameter settings and operated in parallel on the high-performance cloud for large-scale online analysis. The infrastructure/terminal selection in process of CVSimaaS aims to optimally provide resources for the DT based simulation. The optimization model of management and scheduling would be used for the infrastructure/terminal selection, and it could be implemented by the negotiation agents among clouds and edges shown in Fig. 10.

The designer ① develops the DT model and encapsulates the model with its solver and dependent platform into the container image; ② uploads the container image to the closest infrastructure of cloud or edge for integration and application; ③ then registers the container image into the virtual simulation resource pool where blockchain (Christidis and Devetsikiotis, 2016) could be adopted not only to facilitate bookkeeping in the global scope, but also to enhance the traceablility ensuring intellectual property rights.

The operator ④ proposes the requirement for operation of DTs and defines how many instances would be operated for large-scale online analysis; ⑤ could acquire the required container image and its location through search, matching, composition based on simulation resource service; ⑥ would acquire the selected infrastructure and terminal after negotiation by infrastructure services of clouds and edges; ⑦ would acquire the instance of the container

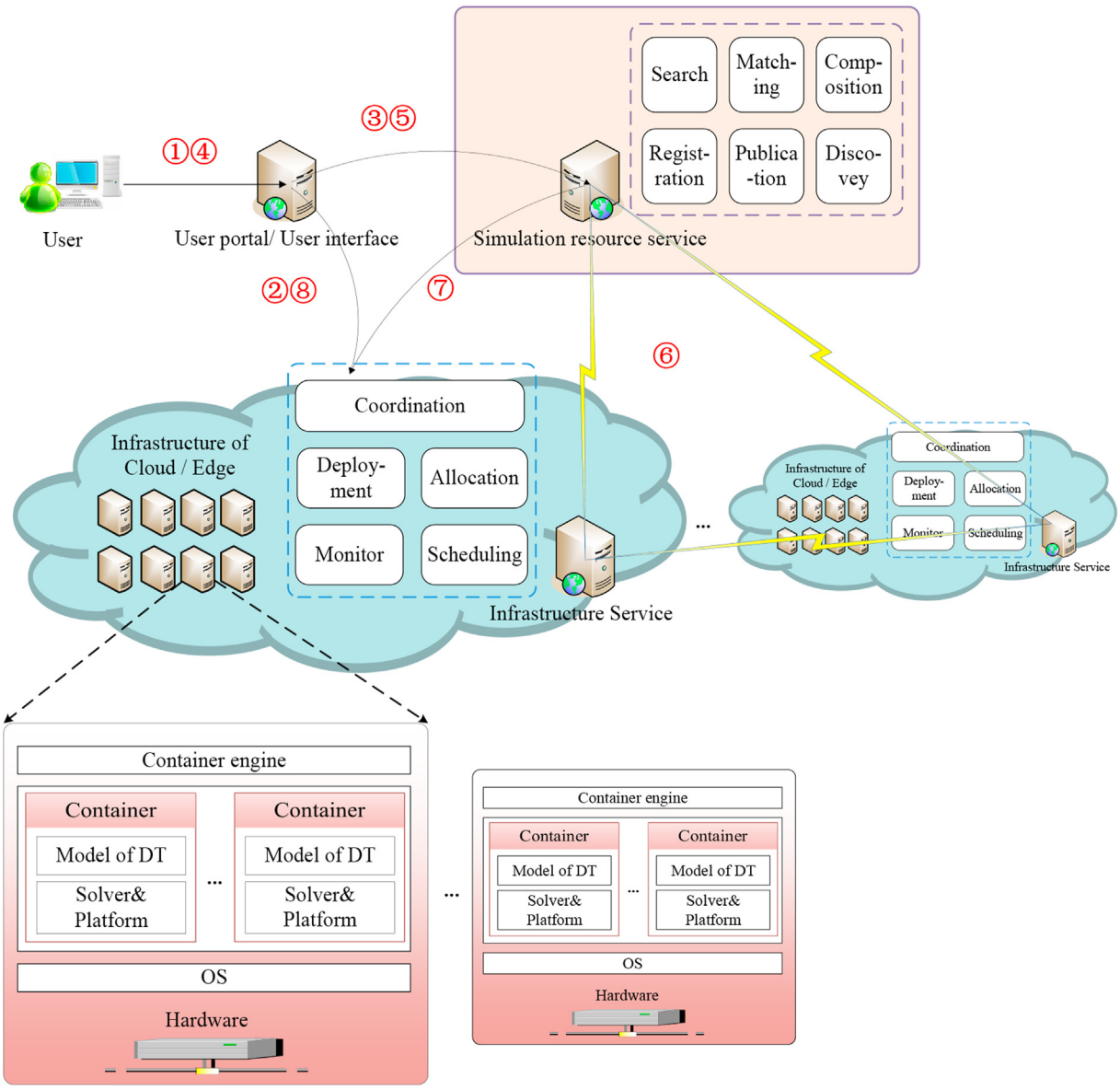


**Fig. 10.** Optimizing model of management and scheduling.

image which runs on the selected infrastructure/terminal; ⑧ ultimately would access the service and implement large-scale online simulation for model training and scheme search.

Security issues are crucial for those SimaaS steps in which human, resources and DT models are highly connected in an open environment. Cyber-attacks can cause serious problems to the DT simulation and the cyber-physical system, through hacking virtual entities and networks or inserting malicious codes in the cyber-space. From the data perspective, the blockchain technology emerges as a novel promising solution that can be explored to increase the data security and privacy (Yang et al., 2020). Artificial intelligence techniques can be used to detect intrusion and attacks with improved accuracy (Saljoughi et al., 2017). We leave those important security issues as future work.

It could be seen from Fig. 10 that the container image is only stored in part of the infrastructure at the beginning. Due to its advantage of convenient and consistent deployment, it could be migrated to the appropriate infrastructure for implement. The main target of the optimizing model is to make decision about which infrastructure should be selected to minimize the total run time which is the QoS of the service. The total run time consists of simulation operation time, migration time and communication time. To minimize the total run time, the optimizing model should schedule as many instances as possible to implement in parallel to minimize the simulation operation time, and should migrate necessarily and appropriately to minimize the migration time and communication time with the physical object which could not migrated meanwhile. Thus, the model should consider parallel computing power and distributed interaction cost, which could refer to multi-centric model of resource management (MMRM) in cloud simulation proposed by author's team earlier (Lin et al., 2015). However, it could be modified and simplified properly for the DT-based simulation need not operate across centers because of the advantage of convenient and consistent deployment. The modified MMRM is shown as follow:

Firstly, we assume that there are $\boldsymbol{M}$ infrastructure (sub centers) and $\boldsymbol{N}$ sub DT models, then the service quality evaluation matrix $\boldsymbol{E}$ could be expressed as $\boldsymbol{E} = [\boldsymbol{F_{ij}}]_{\boldsymbol{M}\times\boldsymbol{M}}$, where $\boldsymbol{F_{ij}} = [\boldsymbol{f_{hk}}]_{\boldsymbol{N}\times\boldsymbol{N}}$ shown in (7).

$$E=\begin{bmatrix} F_{11} & \cdots & & F_{1j} & & \cdots & F_{1M} \\ \vdots & & & \vdots & & & \vdots \\ & & \begin{bmatrix} f_{11} & \cdots & f_{1k} & \cdots & f_{1N} \\ \vdots & \ddots & \vdots & & \vdots \\ f_{h1} & \cdots & f_{hk} & \cdots & f_{hN} \\ \vdots & & \vdots & \ddots & \vdots \\ f_{N1} & \cdots & f_{Nk} & \cdots & f_{NN} \end{bmatrix} & & \\ F_{i1} & \cdots & & & & \cdots F_{iM} \vdots\vdots\vdots F_{M1} \cdots c_{Mj} \cdots F_{MM} & \end{bmatrix} \tag{7}$$

If $\boldsymbol{i}=\boldsymbol{j}$, then $\boldsymbol{F_{ij}}$ is on the diagonal which represents the parallel computing power of infrastructure $\boldsymbol{i}$ for the DT-based simulation shown in (8). In the matrix, all the off-diagonal elements are given 0, and $\boldsymbol{Avail_{ih}}$ represents the availability of the parallel computing power of infrastructure $\boldsymbol{i}$ associated with the above QoS of infrastructure for the sub model $\boldsymbol{h}$.

$$F_{ij}=\begin{bmatrix} \frac{1}{Avail_{i1}} & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & \frac{1}{Avail_{ih}} & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & \frac{1}{Avail_{iN}} \end{bmatrix} \tag{8}$$

If $\boldsymbol{i}\neq\boldsymbol{j}$, then the off-diagonal element $\boldsymbol{F_{ij}}$ represents the distributed interaction cost about the migration of DT-based simulation from infrastructure $\boldsymbol{i}$ to infrastructure $\boldsymbol{j}$ shown in (9). In the matrix, all the off-diagonal elements are given 0, and $\boldsymbol{c_{ih}}$ represents the migration efficiency of the sub model $\boldsymbol{h}$ from infrastructure $\boldsymbol{i}$ which could be associated with the above QoS of a composite image, and the communication efficiency of the sub model $\boldsymbol{h}$ to the physical object near infrastructure $\boldsymbol{i}$ which could be associated with the above QoS of a composite image (we assuming that the physical object could not migrated, and all sub models are migrated from the location of the physical objects).

$$F_{\mathrm{ij}}=\begin{bmatrix} \frac{1}{c_{i1}} & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & & & \vdots \\ 0 & \cdots & \frac{1}{c_{ih}} & \cdots & 0 \\ & & & \ddots & \vdots \\ \vdots & & & & \\ 0 & \cdots & 0 & & \frac{1}{c_{iN}} \end{bmatrix} \tag{9}$$

In addition, infrastructure selection vector $\boldsymbol{X^T}=\boldsymbol{[Y_i^T]_{1\times M}}$ should be set, where $\boldsymbol{Y_i^T}=\boldsymbol{[y_h]_{1\times N}}$ as shown in (10). If infrastructure $\boldsymbol{i}$ is selected, then $\boldsymbol{Y_i^T}=\boldsymbol{I}$, otherwise $\boldsymbol{Y_i^T}=\boldsymbol{0}$.

$$X^T=\left[Y_1^T, Y_2^T, \cdots, [y_1, y_2, \cdots, y_h, \cdots y_N]_i, \cdots Y_M^T\right] \tag{10}$$

So far, the objective function of the optimizing model could be expressed as shown $\boldsymbol{(X^T\times E\times X)/2}$ in (11) and (12).

$$\frac{1}{2}\times X^T\times E\times X=\frac{1}{2}\times\left[\sum_{i=1}^{M}Y_i^T\times F_{ij}\right]_{1\times M}\times[Y_j]_{M\times 1}=\frac{1}{2}\times\sum_{j=1}^{M}\times\sum_{i=1}^{M}Y_i^T\times F_{ij}\times Y_j \tag{11}$$

$$Y_i^T\times F_{ij}\times Y_j=[y_h]_{1\times N}\times[f_{hk}]_{N\times N}\times[y_k]_{N\times 1}=\sum_{k=1}^{N}\sum_{h=1}^{N}y_h\times f_{hk}\times y_k \tag{12}$$

Let's set a parameter $\boldsymbol{c_{max}}$ which means the maximum acceptable distributed interaction cost, and set weighting factors $\boldsymbol{\omega_1}$ and (1 - $\boldsymbol{\omega_1}$) which represents the weight of parallel computing power and distributed interaction cost. Then, we could define a complete optimizing model shown in (13).

$$\begin{aligned} &\text{Minimize } \omega_1\times\left(\sum_{i=1}^{M}Y_i^T\times F_{ii}\times Y_i\right)+(1-\omega_1)\times\left(\sum_{j=1}^{M}\sum_{i=1,i\neq j}^{M}Y_i^T\times F_{ij}\times Y_j\right) \\ &\text{Subject to } \sum_{h=1}^{N}y_h=1,\ y_h=0,1,\ y_h\in Y_i\ \forall i \\ &Y_i^T\times F_{ij}\times Y_j\leq c_{\max}\ \forall i,j\ i\neq j \end{aligned} \tag{13}$$

Obviously, the more available of the parallel computing power and the more efficiency of the distributed interaction cost, the more suitable about the infrastructure selected. The model could be solved by various optimization algorithms, but it is not the focus of this paper.

## 6. Application examples

### 6.1. Requirement of intelligent manufacturing workshop scheduling

The proposed methodology has been applied to an intelligent manufacturing workshop scheduling application in lean manufacturing which is an important kind of smart industrial service system (Nasab et al., 2012). As shown in Fig. 11, the

application (Xiao et al., 2019) consists of two parts, the one is physical object of workshop, the other is DT of workshop. The physical object could be divided into several levels which could effectively collect the status data of the workshop and provide them to the DT. The DT could be divided into simulation model (including parametric model of production line, logistics model and mechanical arm model.) and decision-making model. Based on the status data of the workshop, the decision making model forms scheduling plan based on an improved particle swarm optimization algorithm (Xiao et al., 2016), and loads the scheduling plan into simulation models forming multiple instances of simulation system with different states and random disturbances. In each round of algorithm iteration, based on the instances, the large-scale different simulation experiments would be parallel construct and operated, then feeds back the evaluation result to the decision-making model. The particle swarm optimization algorithm model in the decision-making model would adjust each candidate solution based on the evaluation result, and completes a round of algorithm iteration. After multiple algorithm iterations, the decision-making model generates the best decision result to control the physical object. Then, the result could be implemented and verified in the physical object of the workshop.

In this application, the requirements of the service are shown in Table 1. A parametric model of production line is modeled and simulated with the solver/platform Anylogic (Kondratyev and Garifullin, 2009). A Logistics model is built and simulated with the solver/platform Matlab/Simulink (Bucher and Balemi, 2006). A mechanical arm model is simulated based on the executable program provided by the manufacturer. And a decision-making model is self developed by visual C (VC). All the model would run on Ubuntu OS. Moreover, the container related techniques are applied to complete the construction and operation of the DT-based simulation.

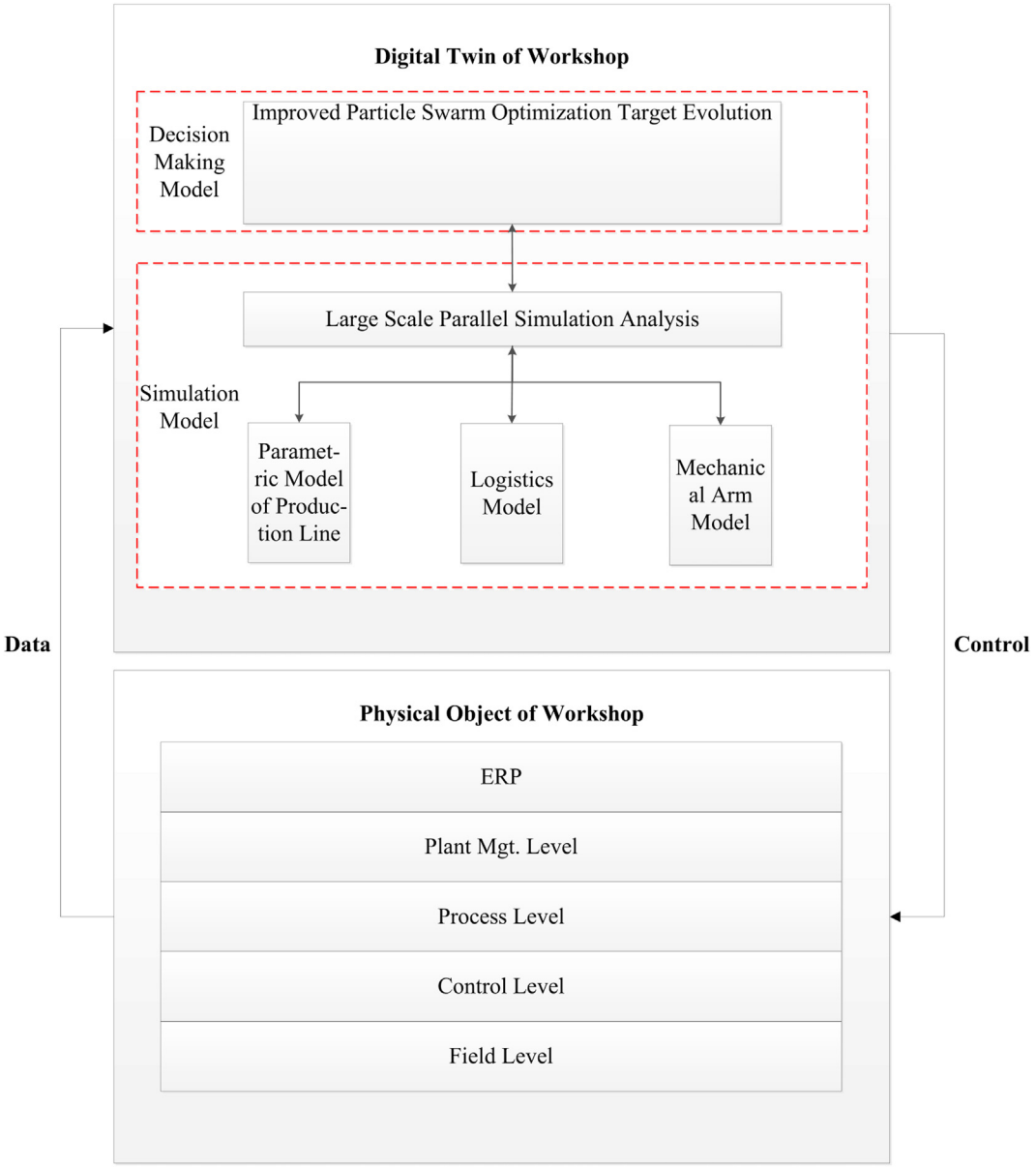


**Fig. 11.** A framework of intelligent manufacturing workshop scheduling application.

**Table 1**
Requirements of the service.

| Model | Solver/Platform | Envvar | OS | CPUNum, MEMSize, NETSpd |
|---|---|---|---|---|
| Parametric Model of Production Line | Anylogic, CERTI | Port:6379<br>Port:6099 | Linux | CPU with 1 core, Memory 1 GB, Bandwidth 200 M |
| Logistics model | Matlab/Simulink, CERTI | Port:6379<br>Port:6099 | Linux | CPU with 1 core, Memory 1 GB, Bandwidth 200 M |
| Mechanical Arm Model | Executable program provided by the manufacturer, CERTI | Port:6379<br>Port:6099<br>Port:5901 | Linux | CPU with 1 core, Memory 1 GB, Bandwidth 200 M |
| Decision-making Model | VC, CERTI | Port:6379<br>Port:6099 | Linux | CPU with 2 core, Memory 2 GB, Bandwidth 200 M |

### 6.2. Implement and application of the CVSimaaS

Some specific details should be dealt with in the implement and application of the CVSimaaS shown in Fig. 12.

(1) Encapsulate an incremental image based on the basic image, and store them into the images repository.
(2) Instantiate the noVNC (Bernstein, 2014) container, and instantiate VNC container additionally depending by the requirement of graphical interaction.
(3) Realize port mapping function through NodePort of service object in k8s (Tan et al., 2010), avoiding port conflicts in large scale simulation.
(4) Access the application by browser or other service invocation so that intelligent manufacturing workshop scheduling application could be operated.

### 6.3. Analysis and discussion of the results

Preliminary application has been carried out in simulated cloud, edge and end environment. Fig. 13 shows the screenshots of the application (Xiao et al., 2019), where Figure (a) is about the container image encapsulation, Figure (b) is the relevant container images list, Figure (c) is about the instantiation of a container image, Figure (d) is the visualization for a DT-based simulation system. It can be seen from the Figure (c) that many containers could run on one node at the same time because their volumes are small.

The quantitative experimental results are shown as follows. As the basic operating system environment is simplified, an average of 70% of the image size compression is achieved (Fig. 14), and an average of 89% speed increase is also achieved in terms of the start-up time (Fig. 15).

As shown in Table 2, the migration (Xu et al., 2020) time consumption of a container image is significantly lower than a traditional VM image, and so does the simulation operation time consumption with the same improved particle swarm optimization algorithm and the same number of instances, particles and iterations. On the one hand, it benefits from faster start-up speed; on the other hand, it also benefits from the simulation model can run on the same node to improve the interoperability efficiency. Benefiting from the advantage of less resource consumption and high efficiency, more instances could be constructed, and the simulation operation time consumption realizes further improvement.

Compared our research results with the similar study about virtualization-based simulation, it is easy to draw the following conclusions. The VM based simulation approach can significantly improve the utilization of computing resources, the efficiency and the flexibility of simulation resource deployment and simulation system construction, compared with traditional simulation conducted on physical (or non-virtualized) computers (Ren et al., 2012). VMs run a complete operating system—including its own kernel and provides complete isolation from the host operating system and other VMs. This is useful when a strong security boundary is critical, but for many cases, it is too heavyweight and consumes much resources, showing the existence of chances to promote green production. In contrast, a container is an isolated, lightweight silo for running an application on the host operating system. Containers build on top of the host operating system's kernel, and contain only applications and some lightweight operating system APIs and services. Thus, the proposed container virtualization-based simulation approach (in which DT based simulation applications run in containers) could not only directly promote green energy conservation by saving the computing and communication resources, but also indirectly help improve the efficiency of industrial assets and processes it simulates, save energy and avoid resource waste.

## 7. Conclusion and future work

This paper proposes a methodology towards container virtualization-based simulation as a service supporting digital twin based advanced analysis on demand. The DT based simulation systems (consisting of DT models and required software

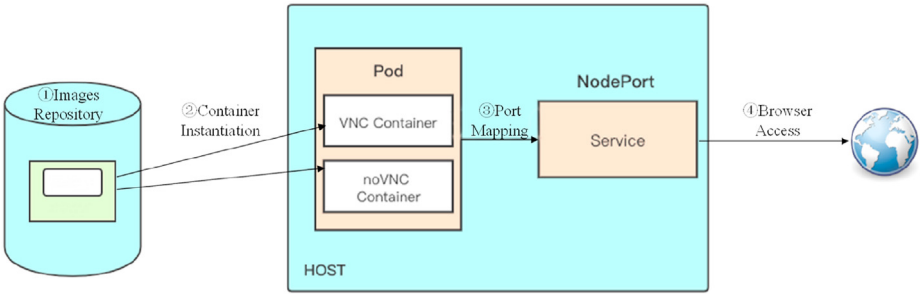


**Fig. 12.** Specific implement and application of the CVSimaaS.

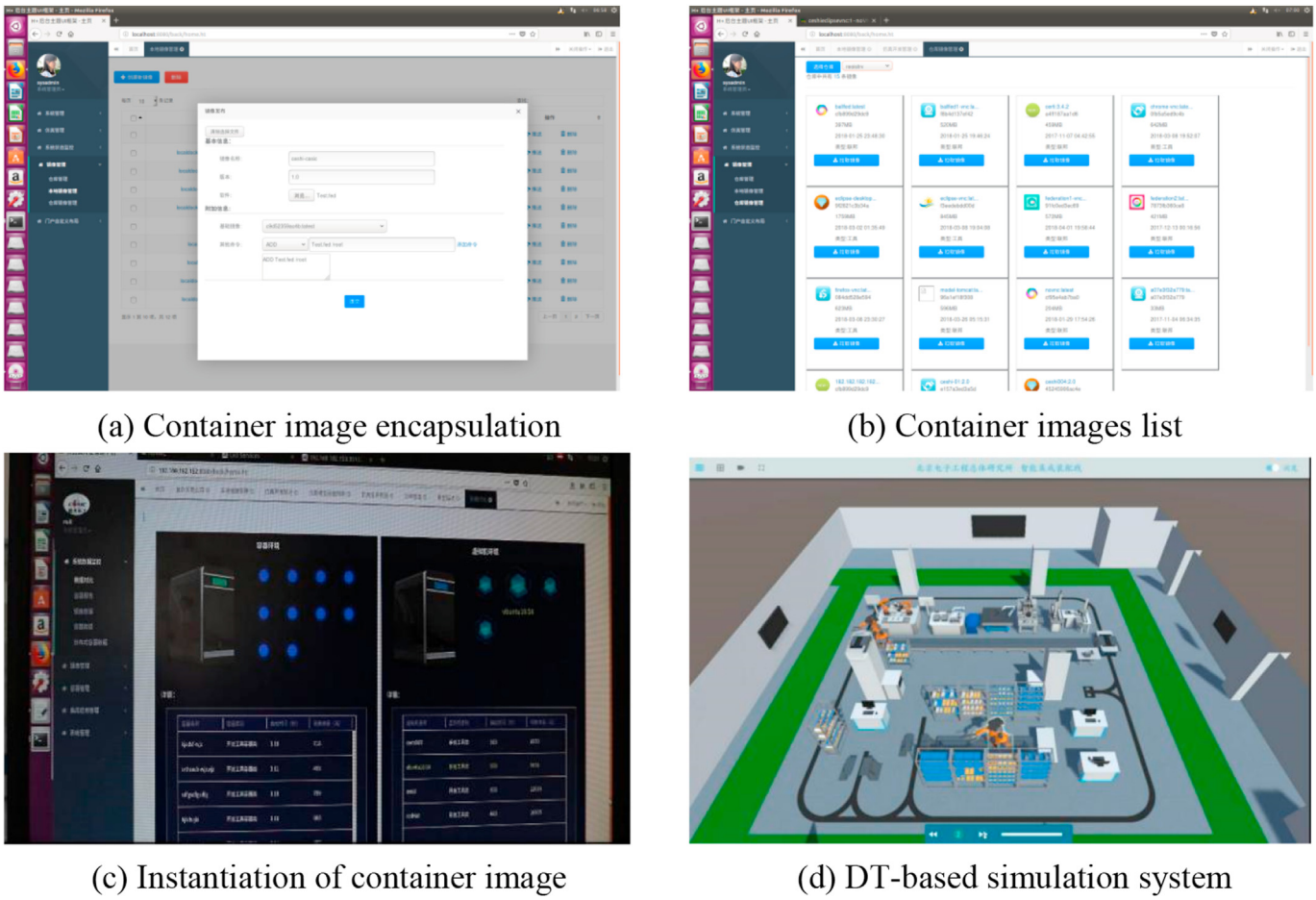


**Fig. 13.** The screenshots of the application.

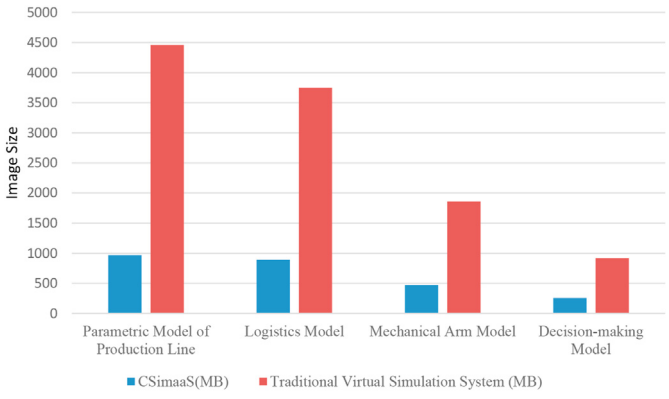


**Fig. 14.** Comparison of image size.

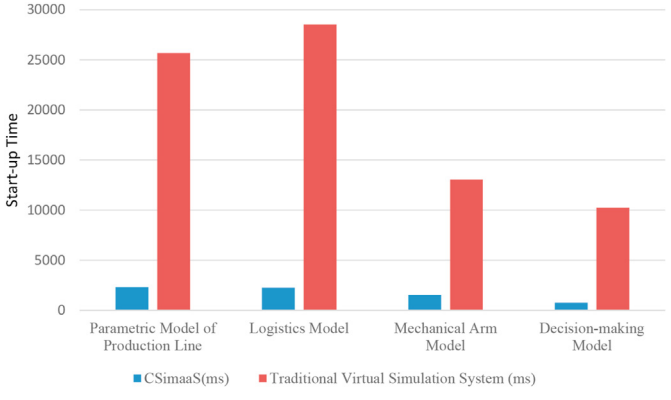


**Fig. 15.** Comparison of start-up time.

environments), mostly deployed and ran in the cloud or on the developers' personal computers, are not well synchronized with the corresponding physical counterparts far away at the network edge in the physical world, thus the real-time data could not be fed back in time to adjust the DT models and make very good decisions via simulation, and the smart decisions could not be rapidly sent to actuate the physical things for better performance. Training the DT models in the cloud and deploying and running the DT based simulation systems at the edge nodes would be a promising way to address the above problems. The DT models and the nearby physical things can mutually optimize each other through real-time dual way communications. The current heavyweight VM based simulation approaches (Ren et al., 2012) consume much storage, computing and network resources and make it hard to transmit the VM template to the edge nodes and run the DT based simulation systems on the edge nodes with limited computing resources. We innovatively use lightweight container virtualization technology to explore a new simulation methodology in terms of paradigm, architecture, approach and resource scheduling model, for efficient large-scale DT simulations on demand. To the best of our

**Table 2**
Comparison of the application results.

| Method | Hardware | Number of Instances | Number of Particles | Number of Iterations | migration time Consumption (ms) | Simulation Operation Time Consumption (ms) |
|---|---|---|---|---|---|---|
| Improved Particle Swarm Optimization Algorithm Based on Traditional Virtualization | CPU with 4 cores, Memory 6 GB, Bandwidth 100 M | 10 | 120 | 100 | 446,531 | 21,600 |
| Improved Particle Swarm Optimization Algorithm Based on Container Operation | CPU with 4 cores, Memory 6 GB, Bandwidth 100 M | 10 | 120 | 100 | 11,256 | 7383 |
| Improved Particle Swarm Optimization Algorithm Based on Container Operation | CPU with 4 cores, Memory 6 GB, Bandwidth 100 M | 20 | 120 | 100 | 11,256 | 4036 |

knowledge, we have not seen such kinds of work.

The main contributions are concluded as the following:

(1) Taking advantage of lightweight container virtualization, a new paradigm - container-based simulation as a service (CVSimaaS) is proposed to support DT-based simulation on the cloud, edge and device on demand, which can expand the application and deployment of DT-based simulation systems to various heterogeneous devices with good elasticity for large-scale online analysis. Using lightweight containers to accommodate DT-based simulation systems, the on demand dynamic deployment of simulation systems can be significantly improved with much higher efficiency.
(2) Based on container virtualization technology, a cloud-edge-device architecture is put forward with a process to support the CVSimaaS paradigm, which upgrades traditional virtual machine based simulation systems to container based simulation systems for dynamic construction and efficient operation of a complex DT-based simulation system. This can be used to guide the future research and implementation of DT based simulation on the cloud and edge computing environment.
(3) To enable search/matching/composition of container images and computing devices, a layered description model for them and a mathematical optimization model of resource management and scheduling are put forward to support the efficient construction of DT-based simulation systems and provide SimaaS to users, promoting flexible sharing of fine-grained computation resource and convenient system deployment.

Our methodology uses the container virtualization as the basis, so one limitation is that containers could not well support the DT models and software environment that run on the Windows OS. More large-scale implementations should be developed to further verify our methodology. Thus our future work includes: first, the latest development of virtualization technology needs to be continuously tracked and introduced to better encapsulate various OS, solver and dependent platform; second, the proposed methodology should be used fully in the industrial Internet supporting the DT-based simulation system deployed on the cloud, edge and device on-demand for large-scale online analysis in parallel; third, new intelligent optimization algorithms should be designed to solve practical resource scheduling problems; fourth, container-based security problems of DT-based simulation for multi-tenants should be studied and strengthened.

## CRediT authorship contribution statement

**Ting Yu Lin:** Investigation, Conceptualization, Methodology, Software, Writing - original draft. **Guoqiang Shi:** Investigation, Conceptualization, Methodology, Software, Writing - original draft. **Chen Yang:** Investigation, Conceptualization, Methodology, Software, Writing - original draft. **Yingxi Zhang:** Investigation, Data curation. **Jiezhang Wang:** Investigation, Data curation. **Zhengxuan Jia:** Investigation, Software, Validation. **Liqin Guo:** Investigation, Data curation, Software, Validation. **Yingying Xiao:** Investigation, Validation, Formal analysis. **Zhiqiang Wei:** Investigation, Data curation, Software, Validation. **Shulin Lan:** Investigation, Conceptualization, Methodology, Software, Writing - original draft.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgement

The research is supported by the National Defense Basic Scientific Research Project, Ministry of Industry and Information Technology, China (No. JCKY2018204C004), the National key R&D Program of China , Ministry of Industry and Information Technology, China(No. 2018YFB1701600) and the Beijing Institute of Technology Research Fund Program for Young Scholars.

## Appendix. Notations

| | | | |
|---|---|---|---|
| DT | Digital Twin | CPS | Cyber-Physical System |
| **CVSimaaS** | Container Virtualization based Simulation as a Service | **VM** | Virtual Machine |
| **SimaaS** | Simulation as a Service | **PM** | Physical Machine |
| **IoT** | Internet of Things | **QoS** | Quality of Service |

## References

Anderson, T., Peterson, L., Shenker, S., et al., 2005. Overcoming the internet impasse through virtualization. Computer 38 (4), 34–41. https://doi.org/10.1109/MC.2005.136.

Barford, P., Blodgett, M., 2007. Toward botnet mesocosms. In: Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets, 6-6.

Barham, P., Dragovic, B., Fraser, K., et al., 2003. Xen and the art of virtualization. In: Proceedings of the 19th ACM Symposium on Operating Systems Principles, pp. 164–177.

Benlian, A., Hess, T., 2012. Opportunities and risks of software-as-a-service: findings from a survey of IT executives. Decis. Support Syst. 52 (1), 232–246. https://doi.org/10.1016/j.dss.2011.07.007.

Bennett, T.R., Gans, N., Jafari, R., 2015. A data-driven synchronization technique for cyber-physical systems. In: Proceedings of the Second International Workshop on the Swarm at the Edge of the Cloud, pp. 49–54.

Bernstein, D., 2014. Containers and cloud: from lxc to docker to kubernetes. IEEE Cloud Computing 1 (3), 81–84.

Bernstein, David, 2014. Containers and cloud: from lxc to docker to kubernetes. IEEE Cloud Computing 1 (3), 81–84. https://doi.org/10.1109/MCC.2014.51.

Blasco, J.M.D., Romeo, A., Heyns, D., Fernandes, J., et al., 2020. The H2020 OCRE project opens the gates of the commercial cloud and EO services usage to the

research community. Emerging Science Journal 4 (2), 89–103. https://doi.org/10.28991/esj-2020-01213.
Bolton, R.N., McColl-Kennedy, J.R., Cheung, L., et al., 2018. Customer experience challenges: bringing together digital, physical and social realms. Journal of Service Management 29 (5), 776–808. https://doi.org/10.1108/JOSM-04-2018-0113.
Boschert, S., Rosen, R., 2016. Digital twin-the simulation aspect. In: Mechatronic Futures: Challenges and Solutions for Mechatronic Systems and Their Designers. Springer International Publishing, pp. 59–74.
Bryant, David, J., 2006. Rethinking ooda: toward a modern cognitive framework of command decision making. Mil. Psychol. 18 (3), 183–206.
Bucher, R., Balemi, S., 2006. Rapid controller prototyping with matlab/simulink and linux. Contr. Eng. Pract. 14 (2), 185–192. https://doi.org/10.1016/j.conengprac.2004.09.009.
Buss, A., Ruck, J., 2004. Joint modeling and analysis using XMSF Web services. In: Proceedings of the 2004 Winter Simulation Conference, vol. 1. IEEE.
Buyya, R., Yeo, C.S., Venugopal, S., et al., 2009. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Generat. Comput. Syst. 25 (6), 599–616. https://doi.org/10.1016/j.future.2008.12.001.
Christidis, K., Devetsikiotis, M., 2016. Blockchains and smart contracts for the internet of things. IEEE Access 4, 2292–2303. https://doi.org/10.1109/ACCESS.2016.2566339.
Derler, P., Lee, E.A., Vincentelli, A.S., 2012. Modeling cyber physical systems. Proc. IEEE 100 (1), 13–28. https://doi.org/10.1109/JPROC.2011.2160929.
GE Digital, 2018. Digital Twin: digitize assets and processes to enable better industrial outcomes [online]. Available from: https://www.ge.com/digital/applications/digital-twin.
Dragoicea, M., Bucur, L., Tsai, W.T., et al., 2012. Integrating HLA and service-oriented architecture in a simulation framework. In: 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, pp. 861–866.
Dua, R., Raja, A.R., Kakadia, D., 2014. Virtualization vs containerization to support PaaS. In: 2014 IEEE International Conference on Cloud Engineering (IC2E). IEEE Computer Society, pp. 610–614.
Fritzson, P., 2014. Appendix G: FMI-functional mockup interface. In: Principles of Object Oriented Modeling and Simulation with Modelica 3.3. John Wiley & Sons, Ltd.
Glaessgen, E., Stargel, D., 2012. The Digital Twin Paradigm for Future NASA and US Air Force Vehicles. 53rd Structures, Structural Dynamics and Materials Conference: Special Session on the Digital Twin, pp. 1–14.
Guo, L., Wang, M., Ruan, C., 2017. A cloud simulation based environment for multidisciplinary collaborative simulation and optimization. CSCWD 445–450.
Karakra, A., Fontanili, F., Lamine, E., et al., 2018. Pervasive computing integrated discrete event simulation for a hospital digital twin. In: 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA). IEEE, pp. 1–6.
Khalid, M.F., Ismail, B.I., Mydin, M.N.M., 2017. Performance comparison of image and workload management of edge computing using different virtualization technologies. Adv. Sci. Lett. 23 (6), 5064–5068.
Kondratyev, M., Garifullin, M., 2009. Parallel Discrete Event Simulation with AnyLogic. International Conference on Parallel Computing Technologies. Springer, pp. 226–236.
Lasnier, G., Cardoso, J., Siron, P., et al., 2013. Distributed simulation of heterogeneous and real-time systems. In: 2013 IEEE/ACM 17th International Symposium on Distributed Simulation and Real Time Applications, vol. 10, pp. 55–62, 1.
Lawton, & G., 2008. Developing software online with platform-as-a-service technology. Computer 41 (6), 13–15. https://doi.org/10.1016/j.jnca.2013.10.004.
Leng, J., Yan, D., Liu, Q., Zhang, H., Zhao, G., et al., 2019. Digital twin-driven joint optimisation of packing and storage assignment in large-scale automated high-rise warehouse product-service system. Int. J. Comput. Integrated Manuf. 1–18. https://doi.org/10.1080/0951192X.2019.1667032.
Leng, J., Liu, Q., Ye, S., Jing, J., Wang, Y., Zhang, C., et al., 2020. Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model. Robot. Comput. Integrated Manuf. 63 https://doi.org/10.1016/j.rcim.2019.101895, 101895.
Li, B.H., Chai, X., 2002. Virtual prototyping engineering for complex product. Comput. Integr. Manuf. Syst. 8 (9), 678–683.
Li, B.H., Chai, X., Di, Y., et al., 2005. Research on service oriented simulation grid. In: Proceedings Autonomous Decentralized Systems. IEEE, pp. 7–14.
Li, B.H., Chai, X., Hou, B., Zhu, W., Mu, S., 2009a. Cloud simulation platform. In: Proceedings of the 2009 Grand Challenges in Modeling & Simulation Conference. Society for Modeling & Simulation International, pp. 303–307.
Li, B.H., Chai, X., Hou, B., et al., 2009b. Networked modeling & simulation platform based on concept of cloud computing—cloud simulation platform. J. Syst. Simul. 21 (17), 5292–5299.
Li, B.H., Chai, X., Hou, B., et al., 2011. New Advances of the Research on Cloud Simulation. Advanced Methods, Techniques, and Applications in Modeling and Simulation, pp. 144–163.
Li, B.H., Zhang, L., Li, T., et al., 2017. Simulation-based Cyber-Physical Systems and Internet-Of-Things, Guide to Simulation-Based Disciplines. Springer International Publishing, pp. 103–126.
Li, B.H., Shi, G., Lin, T.Y., et al., 2018. Smart simulation cloud (simulation cloud 2.0)-the newly development of simulation cloud. Asian Simulation Conference 168–185.
Li, B.H., Lin, T.Y., Jia, Z., et al., 2019. Smart cloud design technology applied to intelligent industrial system. Comput. Integr. Manuf. Syst. 25 (12), 3090–3102.
Lin, T.Y., Li, B.H., Yang, C., 2015. A multi-centric model of resource and capability management in cloud simulation. EUROSIM 555–560.
Lin, T.Y., Yang, C., Gu, M., et al., 2019. Application technology of cloud manufacturing for aerospace complex products. Comput. Integr. Manuf. Syst. 22 (4), 884–898.
Lin, T.Y., Jia, Z., Yang, C., et al., 2020. Evolutionary Digital Twin: A New Approach for Intelligent Industrial Product Development. Advanced Engineering Informatics (Submitted).
Lofgren, B., Tillman, A.M., 2011. Relating manufacturing system configuration to life-cycle environmental performance: discrete-event simulation supplemented with lca. J. Clean. Prod. 19 (17–18), 2015–2024. https://doi.org/10.1016/j.jclepro.2011.07.014.
Malik, A., Park, A., Fujimoto, R., 2009. Optimistic synchronization of parallel simulations in cloud computing environments. In: IEEE International Conference on Cloud Computing. IEEE, pp. 49–56.
Manvi, S.S., Krishna Shyam, G., 2014. Resource management for infrastructure as a service (iaas) in cloud computing: a survey. J. Netw. Comput. Appl. 41, 424–440. https://doi.org/10.1016/j.jnca.2013.10.004.
Marca, D., Mcgowan, C.L., 1993. IDEFO-SADT Business Process and Enterprise Modelling. Eclectic Solutions Corporation.
Nasab, H.H., Bioki, T.A., Zare, H.K., 2012. Finding a probabilistic approach to analyze lean manufacturing. J. Clean. Prod. 29, 73–81. https://doi.org/10.1016/j.jclepro.2012.02.017.
Panetta, K., 2018. Gartner top 10 strategic technology trends for 2019 [online]. Available from: https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2019/.
Ren, L., Zhang, L., Tao, F., et al., 2012. A methodology towards virtualisation-based high performance simulation platform supporting multidisciplinary design of complex products. Enterprise Inf. Syst. 6 (3), 267–290. https://doi.org/10.1080/17517575.2011.592598.
Rufino, J., Alam, M., Ferreira, J., et al., 2017. Orchestration of containerized microservices for IIoT using Docker. In: 2017 IEEE International Conference on Industrial Technology (ICIT). IEEE, pp. 1532–1536.
Sailer, R., Jaeger, T., Valdez, E., et al., 2005. Building a MAC-based security architecture for the Xen opensource hypervisor. In: Proceedings of 21st Annual Computer Security Applications Conference. IEEE Computer Society Press, pp. 276–285.
Saljoughi, A.S., Mehrvarz, M., Mirvaziri, H., 2017. Attacks and intrusion detection in cloud computing using neural networks and particle swarm optimization algorithms. Emerging Science Journal 1 (4), 179–191. https://doi.org/10.28991/ijse-01120.
Shen, W., Yang, C., Gao, L., 2020. Address business crisis caused by COVID-19 with collaborative intelligent manufacturing technologies. IET Collaborative Intelligent Manufacturing 2 (2), 96–99 (online).
Song, W., Kun, W., Hai, J., 2019. Research situation and prospects of operating system virtualization. J. Comput. Res. Dev. 56 (1), 58–68.
Sonnemann, G.W., Schuhmacher, M., Castells, F., 2003. Uncertainty assessment by a Monte Carlo simulation in a life cycle inventory of electricity produced by a waste incinerator. J. Clean. Prod. 11 (3), 279–292. https://doi.org/10.1016/S0959-6526(02)00028-8.
Tan, K.J., Gong, J.W., Wu, B.T., et al., 2010. A remote thin client system for real time multimedia streaming over VNC. In: IEEE International Conference on Multimedia & Expo. IEEE, pp. 992–997.
Trung, T.T., 2020. Smart city and modelling of its unorganized flows using cell machines. Civil Engineering Journal 6 (5), 954–960. https://doi.org/10.28991/cej-2020-03091520.
Tsai, W.T., Li, W., Sarjoughian, H., et al., 2011. SimSaaS: simulation software-as-a-service. Spring Simulation Multi-conference 77–86.
Wang, F.Y., Wang, F.Y., 2010. The emergence of intelligent enterprises: from CPS to CPSS. IEEE Intell. Syst. 25 (4), 85–88. https://doi.org/10.1109/MIS.2010.104.
Wang, X., Yang, L.T., Liu, H., Deen, M.J., 2017. A big data-as-a-service framework: state-of-the-art and perspectives. IEEE Transactions on Big Data 4 (3), 325–340. https://doi.org/10.1109/TBDATA.2017.2757942.
Xiao, Y., Li, B.H., Lin, T., et al., 2016. A self-adaptive shuffled frog leaping algorithm for multivariable PID controller's optimal tuning. In: Asian Simulation Conference. Springer Singapore, pp. 3–16.
Xiao, Y., Wang, M., Guo, L., Xing, C., Zhuang, C., 2019. Intelligent manufacturing plan management based on digital twins. J. Syst. Simul. 31 (11), 2323–2334.
Xu, B., Wu, S., Xiao, J., et al., 2020. Towards efficient live migration of docker containers. In: Submitted to IEEE CLOUD 2020.
Yang, C., Li, B.H., Chai, X., et al., 2012. Cloud manufacturing oriented cloud simulation supporting framework and its application process model. Comput. Integr. Manuf. Syst. 18 (7), 1444–1452.
Yang, C., Shen, W., Wang, X., 2018. The internet of things in manufacturing: key issues and potential applications. IEEE Systems, Man, and Cybernetics Magazine 4 (1), 6–15.
Yang, C., Lan, S., Wang, L., Shen, W., Huang, G.G., 2020. Big data driven edge-cloud collaboration architecture for cloud manufacturing: a software defined perspective. IEEE Access 8, 45938–45950. https://doi.org/10.1109/ACCESS.2020.2977846.
Zeigler, B.P., Praehofer, H., Kim, T.G., 2000. Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems. Academic Press.
Zhang, Y.B., Li, B.H., Chai, X.D., et al., 2011. Research on the virtualization-based cloud simulation resource migration technology. J. Syst. Simul. 23 (6),

1268–1272.
Zheng, P., Lin, T.J., Chen, C.H., Xu, X., 2018. A systematic design approach for service innovation of smart product-service systems. J. Clean. Prod. 201, 657–667. https://doi.org/10.1016/j.jclepro.2018.08.101.
Zhidchenko, V., Malysheva, I., Handroos, H., et al., 2018. Faster than real-time simulation of mobile crane dynamics using digital twin concept. J. Phys. Conf. 1096.