An Agent-Based Modeling and Virtual Reality Application Using Distributed Simulation: Case of a COVID-19 Intensive Care Unit

Jalal Possik , Ali Asgary , Adriano O. Solis , Gregory Zacharewicz , Mohammad A. Shafiee, Mahdi M. Najafabadi, Nazanin Nadri, Abel Guimaraes, Hossein Iranfar, Philip Ma, Christie M. Lee, Mohammadali Tofighi, Mehdi Aarabi, Simon Gorecki, and Jianhong Wu

Abstract—Hospitals and other healthcare settings use various simulation methods to improve their operations, management, and training. The COVID-19 pandemic, with the resulting necessity for rapid and remote assessment, has highlighted the critical role of modeling and simulation in healthcare, particularly distributed simulation (DS). DS enables integration of heterogeneous simulations to further increase the usability and effectiveness of individual simulations. This article presents a DS system that integrates two different simulations developed for a hospital intensive care unit (ICU) ward dedicated to COVID-19 patients. AnyLogic has been used to develop a simulation model of the ICU ward using agent-based and discrete event modeling methods. This simulation depicts and measures physical contacts between healthcare providers and patients. The Unity platform has been utilized to develop a virtual reality simulation of the ICU environment and operations. The high-level architecture, an IEEE standard for DS, has been used to build a cloud-based DS system by integrating and synchronizing the two simulation platforms. While enhancing the capabilities of both simulations, the DS system can be used for training purposes and assessment of different managerial and operational decisions to minimize contacts and disease transmission in the ICU ward by enabling data exchange between the two simulations.

Index Terms—Agent-based modeling (ABM), cloud computing, COVID-19, discrete event simulation (DES), distributed simulation (DS), healthcare systems, high-level architecture (HLA), hybrid simulation, intensive care unit (ICU), interoperability, virtual reality (VR).

I. INTRODUCTION

HOSPITALS and healthcare settings, as with many other application areas, could benefit from different modeling and simulation (M&S) methods and techniques to enhance and optimize their operations and management as well as better train their existing and new staff. The COVID-19 pandemic and the need for rapid assessment of situations and decisions regarding disease prevention, while considering the risks of direct and field-based observations and analysis of various procedures and policies, have amplified the critical role that new simulation and visualization technologies, including virtual reality (VR) and digital twins (DTs), can play in management and training. Earlier research article (e.g., Punnakitikashem et al. [1], [2]) have shown that analysis of healthcare facility operations may lead to the formulation of complex mathematical models. Such models may require the application of complicated algorithms, heuristics, or simulations, which often are not easily understood by, and straightforward for, all the stakeholders in healthcare decision making and operations. In response to this context, attempts are being made to enable simulations to be more accessible and interactive by integrating them with various technologies, including DT, VR and augmented reality (VR/AR), and distributed simulation (DS).

DTs are currently being developed to guide physical/human operations and maintenance of systems and service processes based on data collected in the physical world. With the use of a DT, managers and trainees can virtually examine, assess, and understand the potential impacts of possible situations and

decisions. This will be especially important for building and operating new hospitals, as well as using hospital capacity more cost-effectively and improving resource planning. In addition, the DT will also be able to help identify resource needs and, against this, reveal the professional communities and their social networks that can work on such needs [3]. The professional communities can be not only existing networks but also underlying networks of new potential resources to be explored [4], [5].

While many standalone simulations, VR, and DT applications have been developed to address management and training needs of healthcare settings during the pandemic, integration of these simulation approaches can further increase their usability and effectiveness. The integration of multiple simulation components could be enabled by the use of DS, which would make the components interoperable and provide end users, less familiar with complex M&S environments, with a holistic view of the system and allow them to better understand and use the models and simulations. DS with VR/AR will enable users to visualize the simulated operations, investigate changes in the system, and interact with the running simulations. Direct access to healthcare settings may pose challenges, or even danger due to possible risks such as COVID-19 transmission, in conducting investigations and experiments. To avoid disruption of vital health services, as well as to allow participation/involvement of trainees and managers who do not have physical access to facilities due to distance, DSs can be used as an alternative way to be virtually immersed in such environments. Moreover, conducting a large number of experiments, involving various policy and operational scenarios, using computer-based M&S is incomparably more practical, more cost-effective, and less time-consuming than actually implementing them in real-world settings. While M&S in the healthcare domain is widely used for 1) program development, 2) testing operational procedures, and 3) introducing new medical programs and various training activities [6]–[8], the emergence of new technologies in simulation and visualization provides opportunities for maximizing their use for these functions and activities.

The main purpose of the current article is to integrate a high-fidelity simulation of a hospital intensive care unit (ICU) ward (dedicated to COVID-19 patients), developed using agent-based and discrete event modeling methods, with a VR three-dimensional (3-D) model. This would enable users to interact with the simulation and examine/review different scenarios in the ICU and their impacts. The remainder of this article is organized as follows. Section II provides a review of relevant existing literature and recent research article. Section III describes the DS system architecture and methodology. Section IV presents the DS case study, including the simulation models in AnyLogic and in Unity as well as their integration. We discuss simulation results in Section V. Finally, Section VI concludes this article.

II. Background and Literature Review

There has been an observed exponential increase in the literature involving the development of mathematical models aimed at informing healthcare operations management and decision-making. This is evident from a recent review of such articles published in the *Manufacturing & Service Operations Management (M&SOM)* journal conducted by Keskinocak and Savva [9]. They found that while there were only two such modeling papers in the journal's first 10 years (1999–2008), a total of 48 papers—addressing challenges and decisions faced by various healthcare system players including policymakers, healthcare delivery organizations, and medical professionals—were published in the next 10 years (2009–2018), representing 11.6% of all *M&SOM* papers during that second decade. As the world entered the global pandemic situation and with the urgent need for technological solutions for response at all levels including healthcare facilities, a significant number of researchers focused on innovative approaches to meet these needs. Gao and Sunyaev [10] and, particularly within the context of the COVID-19 pandemic, Yu et al. [11] and Bag et al. [12] have examined the value of some of these solutions built based on big data analytics, artificial intelligence, cloud computing, simulation, and other technological approaches in improving responsiveness and resilience of hospitals and healthcare systems.

One important emerging trend during recent years is the development of hybrid simulations that combine different modeling methods, such as discrete event simulation (DES), agent-based modeling (ABM), and system dynamics (SD) [13], [14]. For instance, Solis et al. [15] and [16] developed a sequential hybrid M&S framework involving two different simulation models running on separate platforms: 1) an Incident Generation Engine, which simulates the "arrival" of emergency incidents and 2) a Response Simulation Model. The first model is a DES model using CPNTools 4.0, generating inputs for the second model, which is an ABM developed using AnyLogic. Ruiz-Martin et al. [17] proposed a hybrid model combining ABM, discrete event system specification, network theory, and Monte Carlo simulation to study communications in emergency plans.

In cases where two simulations may need to be integrated so that bidirectional communications between the two or more M&S components are essential, DS can play a key role. The DS can integrate several heterogeneous simulators operating on different simulation environments [18], [19]. DS systems are becoming more and more complex in terms of both the level of dynamism and the level of heterogeneity within the system. Blair et al. [20] identified four main levels of heterogeneity which may pose barriers to integration and interoperability as follows:

1) Data heterogeneity—different models represent the data in different formats.
2) Middleware heterogeneity—a wide range of middleware to choose from, which would require interaction protocols.
3) Application heterogeneity—different ways to implement programs.
4) Nonfunctional heterogeneity—consideration of nonfunctional properties.

While the data, middleware, and application heterogeneity barriers may be overcome so that integrated simulations can functionally interoperate, a solution cannot be considered to have achieved full interoperability if it does not satisfy the nonfunctional requirements of each of the endpoints. For instance, two systems may have different message delivery requirements or may employ different security protocols, which must be maintained under the interoperability solution [21]–[24].

The HLA standard originally developed by the Department of Defense (DoD) in the United States (US) is considered to be the most popular standard mainly used for solving the interoperability and reusability challenges in DS [21]. The Defense Advanced Research Projects Agency took a decade to develop HLA in the 1990s, resulting in the US DoD 1.3 HLA standard [25]. In the year 2000, it was adopted by the IEEE and called HLA IEEE 1516. It was modified and updated in 2010 to encompass improvements and became known as HLA evolved. Under development is the next version of HLA, to be called HLA 4, which will have new object modeling opportunities as well as new simulation security-related features [26].

HLA is a standard that helps in the development of DS and operates through the creation of a system composed of different simulation components. These components are called "federates." A federation consists of a set of federates, a run-time infrastructure (RTI), and a federation object model (FOM). The RTI provides a standardized set of services for information and data exchange, synchronization, and the management of the federation. The FOM defines the object and interaction classes used to exchange data. The main goal of HLA is to enable simulation interoperability and reuse of simulations running on separate computers having different operating systems, implemented in different programming languages, and distributed on local or wide area networks. All those heterogeneous components are connected into the same federation. Using the publish/subscribe (p/s) mechanisms of HLA and based on the FOM, those different heterogeneous components, called federates, can exchange data and information. HLA is now a well-established IEEE standard used by many researchers and engineers to solve their interoperability issues in the industry [27]–[30]. The DS domain is now becoming more efficient and powerful with web-based environments. Indeed, more and more simulation platforms are becoming available on the cloud while organizations are using more cloud services to reduce the cost and efficiency of their applications. As a result, cloud computing extends the possibilities of implementing the DS technologies that will be easily accessible by the simulation community. Precisely, HLA enables DS to be created in the cloud, making the integration of different simulations easy and fast.

As stated by Macal [31], ABM has characteristics and capabilities that go beyond those of more traditional simulation techniques—e.g., SD and DES. ABM models agents individually, with the diversity in agents' attributes and behaviors giving rise to the system's overall behavior [32]. Recently, Choudhary et al. [33] have applied ABM to assess the resilience of a supply chain network in the face of disruptions since the onset of the COVID-19 pandemic. In the healthcare sector, Tofighi et al. [34] and Possik et al. [8] have used a combination of ABM and DES methods to study the operations of a large hospital hemodialysis unit and to examine outbreak scenarios in the context of COVID-19. With the growing trend toward the use of ABM in general, and in healthcare in particular, it has become increasingly important and necessary to understand and address some of the key challenges in developing a DS system in which ABM and other simulation methods are applied and combined. Large agent-based models coupled with other simulations would require more CPU power and memory than classical SD or DES models [35], especially when the models involve large populations of agents with complex individual behaviors. The emergence of these complex models calls for the integration of distributed technologies into the simulation field as a solution to split the simulation load into different components and run such components on connected machines.

Aside from the aforementioned, VR is an interesting technology that enables individuals to interact effectively with a real-time 3-D computer-managed environment [36]. DES, ABM, and VR techniques have existed in relative isolation until now. Very limited articles exist on integrating these technologies to form simulation-based virtual environments instead of using a user-configured virtual environment. Turner et al. [37] suggested potential opportunities for DES/VR technology combination as well as real-time integrated communication methods and system design concerns. Huang et al. [38] proposed a synthetic design approach to build an integrated ABM/VR environment. To our knowledge, there is currently no practical implementation integrating DES, ABM, and VR. In this article, based on the HLA standard and its mechanisms, we have designed and developed a novel cloud-based DS system that integrates DES and ABM into VR environments. This combination elevates VR to a new level.

Furthermore, we have resolved data, middleware, application, and operating system heterogeneities among three major M&S applications in this article. Interoperability issues between these heterogeneous components were resolved by creating an HLA interface for each of them, which allows them to be integrated and viewed as a single powerful system with subsystems that are completely homogeneous and time-synchronized thanks to this interface.

Consequently, the execution of decision support systems can rely on DS components, including DES, ABM, and VR engines that can run on multiple processors connected through a wide network. The different components will be part of a higher level architecture that may be viewed as one integrated simulator for healthcare operations management. An ICU case study has been successfully implemented on this DS system.

III. DISTRIBUTED SIMULATION COMPONENTS AND ARCHITECTURE

To address some of the above listed challenges and examine possible alternatives, we present a DS system developed to integrate the AnyLogic simulation software [39] with Unity. The HLA DS protocol, discussed in the preceding section, was used for the communication and data exchange between these two heterogeneous components. In this article, we have used Unity3D as an engine synchronized with the process flow of AnyLogic, orchestrated by Papyrus to display the real-time 3-D animation of the integrated simulation in VR format. The AnyLogic engine is written in Java, and all developed models are fully mapped to Java code. However, Unity is mainly a C++ based game engine, and all codes are written in C#, JavaScript/UnityScript, and sometimes in Boo. To solve the interoperability issue, an upper

layer has been added for both AnyLogic and Unity applications. The second part is the integration of Papyrus to the components.

This article builds upon on an earlier attempt by Possik et al. [8] to apply the proposed DS to integrate modeling and visualization of COVID-19 transmission in a hemodialysis unit to optimize the space and equipment arrangement. The current article goes further by developing a DS system for an ICU ward, with active COVID-19 patients, and provides critical support for epidemiological transmission management and training through a VR application.

A. HLA as an Orchestrator for Platform Components

The use of HLA broadens our horizons and opens the door to the development of multiple models operating on different platforms, data exchange, interoperability, reusability, and communication with external systems. The DS system's architecture, development, and simulation synchronization process are presented in this section.

In this article, we use HLA to integrate and exchange data between AnyLogic and Unity. Both components are connected to Papyrus, an open-source, industrial-grade, model-based engineering tool. Thus, the developed DS is composed of three main components: 1) AnyLogic, for ABM development and implementation; 2) Unity, for 3-D development and VR implementation; and 3) Papyrus, as a general modeler and orchestrator engine. The fact that we can now have VR experience with process/agent-based and DES distinguishes our research article from other articles in the literature. We seek to create a sophisticated DS system using the IEEE HLA standard while making use of the benefits of two separate applications. Some simulations, such as process/agent-based and DESs, are not easily possible in Unity. On the other hand, some simulations such as particle modeling that are needed for visualization of particles, such as in disease transmission, are not feasible with DES. However, the embedded particle system in Unity allows for such simulations. One of the key challenges to overcome throughout this research project was the interoperability between AnyLogic and Unity, two separate, heterogeneous applications. The developed DS paves the way for real-world scenarios to be tested in a VR environment while live results from a concurrently running agent-based discrete event simulator are displayed.

B. DS System Architecture

All data are sent or received as objects/attributes or interactions/parameters between the three implemented main components. There exists a common FOM extensible markup language file that lists all shared objects/attributes and interactions/parameters. Papyrus is referred to as a "Master" federate in our article. The Master federate is integrated with the other components through its built HLA interface. Based on the publish mechanism of HLA, the Master federate sends the data input to the connected federates and, based on the HLA subscribe mechanism, receives the output data from the connected federates. Likewise, all connected federates can p/s to the required data. The Java library of the Pitch pRTI [40] is used for the HLA development process.

An HLA interface is developed for each of the components. By implementing an HLA interface to those components, any of them will have both possibilities: creating a federate and creating/joining a new/existing federation. The federation development approach avoids the early phase errors and reduces the cost and time of simulation development. The time management mechanism of HLA is not implemented in this system. All federates in the current system are neither time-regulating nor time-constrained. Therefore, messages between federates are delivered right after the publishing process of any of the connected federates. As shown in Fig. 1, the federates are implemented on different physical or cloud servers, all connected and time-synchronized by the same RTI server. A similar Java code is developed to implement the HLA interface of the Java-based systems (AnyLogic and Papyrus). However, the programming language used in Unity is C#. Consequently, the development of an HLA interface for Unity becomes more complex. IKVM [41], a .NET implementation of a Java Virtual Machine, is used to enable the interoperability between Java and .NET. DLL files are exported from the Java HLA code developed to make Unity HLA compatible. Those DLLs are imported, along with the IKVM.OpenJDK.Core, into the reference libraries of the C# code. After this step, the methods written in Java are used in the C# code (create or join a federation, p/s data, send/receive interactions, etc.).

We used Microsoft Cloud Services, specifically Microsoft Azure [42], to make the RTI available online for all developed federates. No additional server roles or features were installed. Then, we installed the Windows version of Pitch pRTI [40]. The Azure server is used as a central RTI server that manages the federates' connections and data exchange. Each of the federates (AnyLogic, Unity, and Papyrus) is installed on a private server with a public IP address to make it online and available, as a federate, to connect to the RTI installed on Azure.

The RTI services are defined in each of the HLA interfaces shown in Fig. 1. The RTI consists of an executable called central RTI component and a library used by the federates called local RTI component. In our article, HLA services are provided by the Java pRTI library. Calls to an instance of the RTI Ambassador class are used to invoke services in Java APIs. Callbacks, which are supplied through calls to an instance of the Federate Ambassador class, are used by the RTI to deliver information to a federate. Each time a federate updates an object using the *updateAttributeValues()* method, an RTI callback is issued to update all linked federates using the *reflectAttributeValues()* method, leveraging HLA's p/s mechanisms. The FOM file defines the objects and interactions classes that are shared between connected federates. AnyLogic, which runs the ABM and the DES, publishes the process-based simulation objects to the RTI and is also subscribed to various Unity objects, such as particle system data. Unity federate, which runs the 3-D VR environment, is subscribed to AnyLogic's objects, such as the agent type and position. Papyrus publishes interactions to the RTI, such as *runSimulation(), pauseSimulation(), stopSimulation(), runComponent(), pauseComponent()*, among others.

Papyrus is an open-source graphical editing tool based on systems modeling language and unified modeling language (UML)
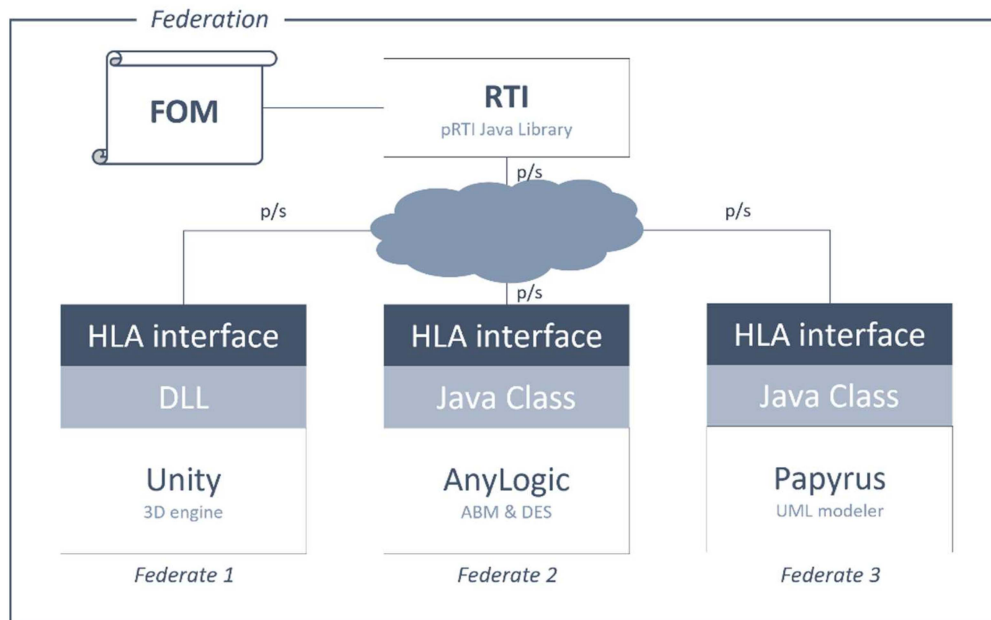
Fig. 1. Cloud-based DS system components.

standards that provides an execution environment, thanks to the foundational UML subset (fUML) standard. This software provides a user-friendly M&S environment for editing UML diagrams. Papyrus provides the ability to manipulate UML profile mechanisms that allow defining extensions over the basic UML standard. Upgrading UML metamodel allows, on the one hand, to customize modeling language for a specific domain, and, on the other hand, to increase the reliability and flexibility of the M&S tool.

Papyrus also provides an open-source UML execution engine, namely Moka, as a plug-in that is compatible with Papyrus and the fUML standard. Moka is integrated into the Papyrus debugger and thus can provide control and animation functions during model execution. It can also be extended to fit with the custom UML profile defined by the user to implement custom behaviors. Among different process modeling tools, Papyrus is selected because of its simplicity and capabilities to communicate with HLA-compatible applications [43]. In this article, Papyrus maintains the simulation orchestration. UML models designed under the tool are used as DS scenarios including risks. Each link with external components (through HLA) is managed by a UML.

Thus, a UML profile is defined for managing the interactions that Papyrus can have with other components. The HLA UML profile is an extension to the UML metamodel. It implements new parameters to UML action components. It also contains the necessary information that will be communicated as objects or interactions for launching the DSs.

With each UML profile comes a Moka extension developed for adding custom behavior to the profile. During the execution phase, the user adds a specific UML profile on every task according to what is expected from it. When the task launches a federate, the user adds an HLA UML profile on the task.

When the specific task is executed by Moka, a custom behavior implemented by the developer is executed. In our case, it represents the connection and communication with the HLA RTI standard.

IV. DISTRIBUTED SIMULATION CASE STUDY

The simulation platform and model are based on the ICU located at Toronto General Hospital (TGH), which is part of the University Health Network. This unit has 30 beds and provides care to approximately 1000 patients annually. This ICU ward has been allocated to COVID-19 patients since the start of the pandemic and is expanded or reduced based on COVID-19 patients' needs. Our DS system integrates two major simulations of this ICU ward. The first simulation is developed in AnyLogic to simulate the ward operations and to analyze disease transmission possibilities based on contacts made and COVID-19 protection measures taken. The second simulation was developed in Unity to be used for VR-based management and training purposes. This section describes these two simulations and their integration using the DS architecture as described in Section III.

A. ICU Ward Simulation in AnyLogic

The multimethod (agent-based and discrete event) simulation model in AnyLogic includes a physical layout, agents, and logical components that define procedures and routines for the agents and represent policy choices regarding daily operations and infectious disease control. Fig. 2 exhibits the physical layout of the ICU ward, presented in a two-dimensional (2-D) window, which includes a few dashboards regarding the status of the model (e.g., shifts and schedules for the agents) and statistical summaries.
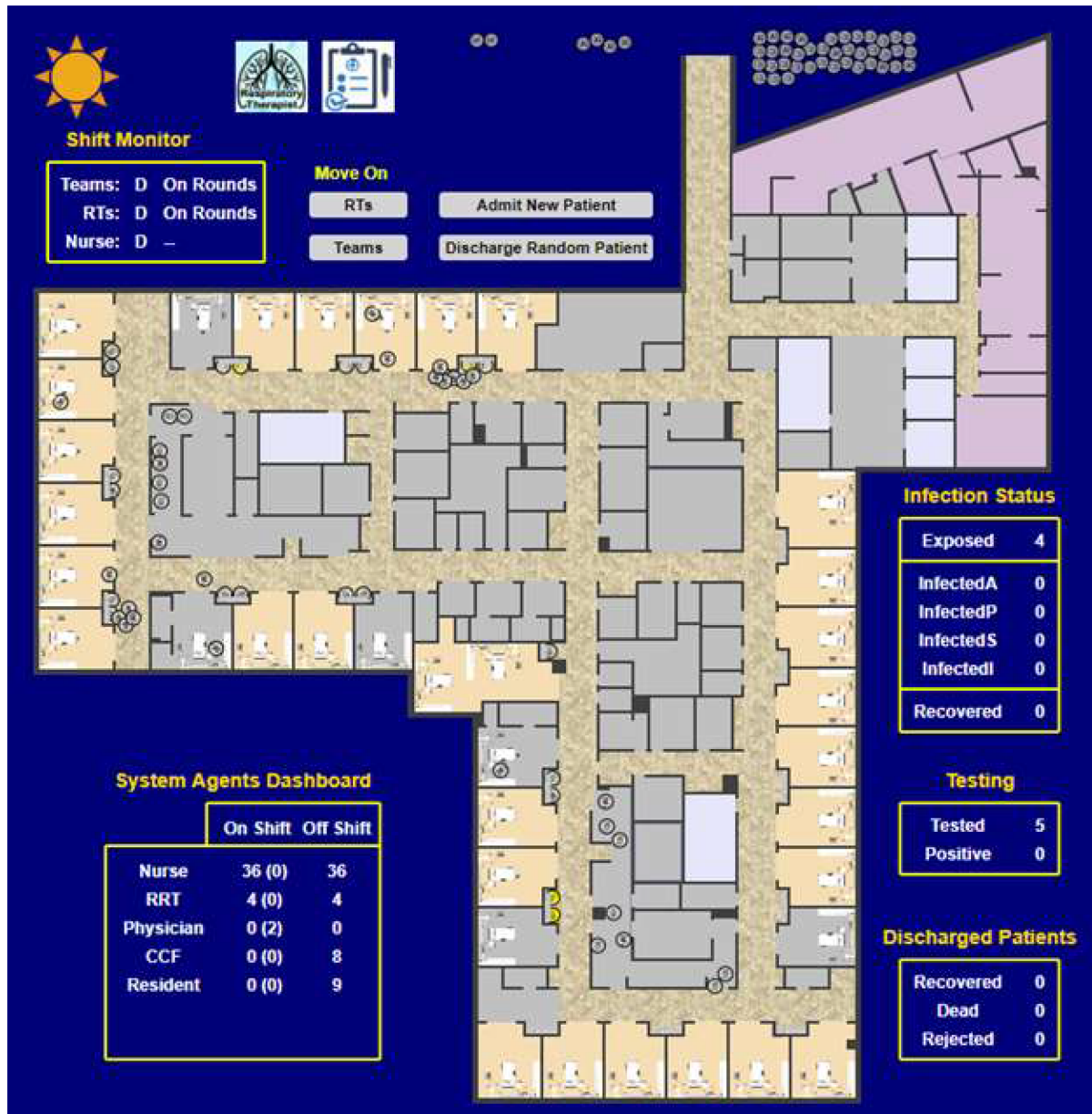
Fig. 2.   2-D model of ICU ward, developed with AnyLogic.

1) Agents: The human agents in this model are the patient and various healthcare providers such as nurse, respiratory therapist (RRT), doctor, critical care fellow (CCF), resident, and allied health staff (AHS). Patients have properties, such as gender, length of stay (at the ICU), whether they need acute care in the ICU, whether they require high-risk procedures (from the viral transmissibility standpoint), etc.

The nurse agent represents the critical care nurses who are assigned to patients on a one-on-one basis. The nurse is assigned to the patient as soon as the patient is admitted. Each of these nurses will have several services scheduled in their active shift for their assigned patient. The RRTs divide the available patients among themselves during their active shift, and each is responsible for several patients. Teams of physicians (doctor, CCF, and resident) are formed to conduct the checkups on their assigned patients.

The grouping features of AnyLogic in the pedestrian library have been utilized to allow these agents to form a team. Other human agent types have limited functionality in this model, as their roles are not centric to the ICU processes and procedures that are captured by this model.

2) ICU Daily Operation: Along with the agents in the simulation model, there are operational processes that each agent follows. This includes the daily routines and ad-hoc services provided to the patients corresponding to their specific needs. There are also some rules or unwritten regulations that put constraints on the modeled environment. Examples of the implemented ICU processes and constraints in this model are as follows.

Each agent type follows a different timetable for shifts and schedules. For example, while nurse and RRT agents have regular 12-h shifts, the team of physicians (doctor, CCF, and
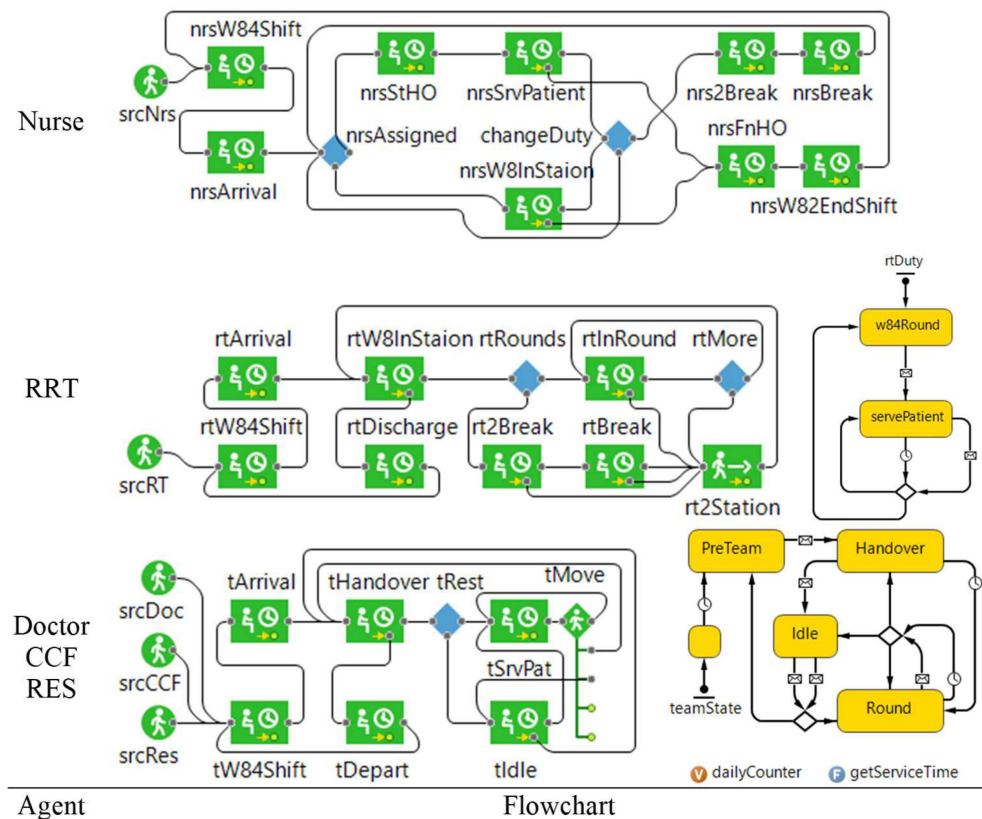
Fig. 3. Workflow of selected active agents in AnyLogic model.

resident agents) follows a shorter day shift, which has a heavier schedule, and a few members of the teams will be on long shifts, requiring them to stay overnight for up to over 24 h. But there are similarities as well. For example, as the agents' shift terminates, the agents from the next shift come into the ICU environment, and the two types of agents (from the previous shift and from the new shift that has just started) will go through a handover process through which the most recent developments in the patients' situations are discussed. Agents' logic is implemented through flowcharts and statecharts.

Critical care nurses are assigned to patients at the beginning of each shift, on a one-on-one basis. The nurses do a handover at the beginning and at the end of their shifts with the nurse from the outgoing or incoming shift. For most of the time, the nurses are in their seats outside of the patient's room, with a window to their assigned patient's room through which the nurse monitors the patient. The nurse agents move in the physical layout according to their given schedule and the workflow presented in top portion of Fig. 3. Four RRTs per shift are assigned to one-fourth of the admitted patients. In each shift, the RRTs have three regular rounds of patient checkups. The RRTs move in the physical layout according to their schedule and the workflow shown in the middle part of Fig. 3. In addition to the operations flowchart through which all the RRTs move around, each RRT also saves its internal state using an individual statechart. This statechart helps each RRT instance to follow a distinct schedule for checkups on assigned patients.

Three types of agents—doctors, CCFs, and residents—form the physician teams. In each shift, two physician teams are formed, and they go on rounds twice per shift. The second of the physician rounds in the day shift is the time in which specific procedures are performed. The team of physicians follows the workflow presented in the bottom portion of Fig. 3, which covers all day and night patient checkup rounds and procedures. At the beginning of the day shift, three members of the day teams will be randomly selected (two CCFs and one resident) to be on a long shift. At the end of the day shift, all team members (apart from the three physicians on long shift) will leave the ICU environment, and the long shifters will move to their idle station and wait until their first service of the night. They follow the night team schedule for the night shift. Once the night shift is over, the three night-time physicians will hand over their shift responsibilities to the new day-shift physicians. Following morning handover, the night-time physicians, who were on their long shift, will leave the ICU environment and will not return to the ICU for at least 48 h.

3) Disease Transmission Module: We have used a hierarchical structure in defining human agents in the model, which allows expanding human properties among the different types of agents (healthcare workers and patients). We have created a super-agent capturing human characteristics, which will be inherited in the human-type agents that exist in the model. One of the components in this super-agent deals with disease transmission.
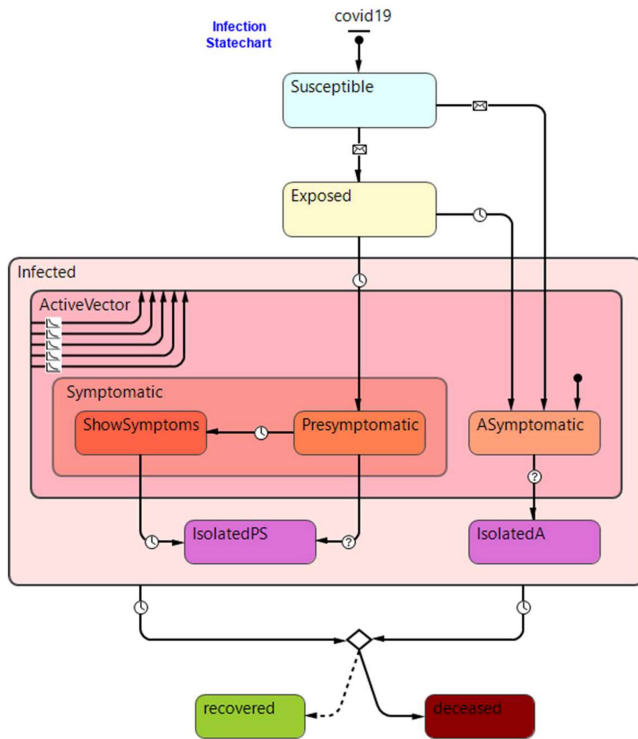
Fig. 4.   Agent-based disease transmission model.

Fig. 5.   ICU room in the training application in Unity.

Fig. 6.   Simulated patient in the VR–ICU application.

In the agent-based disease transmission module, depicted in Fig. 4, we have created an extended version of the popular susceptible, exposed, infected, and recovered model (SEIR) [44], [45]. In our modified model, asymptomatic and presymptomatic infected healthcare workers become new sources of viral contamination and infection, spreading the disease during risky close contacts with other human agents who are in the model's physical space and are susceptible to the infection. Similar models have been adopted in other simulations and were proven to be effective [46]. A state chart in the human super-agent tracks the different phases of infection, including the phases in which the human agent becomes a source of new contamination and disease transmission (ActiveVector state in Fig. 4). In the model, the symptomatic healthcare workers are tested and, having tested positive, they are assumed to be isolated, not being able to transmit the virus anymore. Because symptomatic healthcare workers are assumed to be identified and replaced for the duration of their quarantine or recovery, they are not considered active viral vectors. For the admitted ICU patients, the virus transmission component is constantly active due to our initial assumption that the ICU ward under study only admits COVID-19 patients. The healthcare workers, who get into the ActiveVector state, will also be in the same situation, and both of these are handled by the same sate chart in the super-agent, as described earlier. Because there are several sources of viral contamination in this model, we also record the source of the contamination for the exposed/infected medical staff agent to help identify the most active sources of viral spread.

The model records effective agent contacts (above 5 min in proximity of closer than 2 m) and summarizes them in the form of a contacts matrix as explained in the next subsection. The disease transmission module uses this matrix as input and propagates the infection on that basis [47]. The way the contacts are recorded is that every agent's surroundings within the specified 2-m radius (consistent with the WHO's definition of risky contacts) is checked for other human agents every 1 s. A list of available agents in that proximity is then populated and kept up to date for each time-step during the modeling timeframe. Any of those below 2-m contacts lasting over 5 min is then considered a risky contact and summarized into the effective contacts matrix.

4) Physical Contacts Calculator: Close physical contact among individuals has been identified to be one of the major factors in the transmission of COVID-19 [48]. It is important to count and evaluate contacts between caregivers and patients, as well as among the caregivers themselves. Using the simulation approach and implementing in the current model, a contact calculator method presented by Najafabadi et al. [47] and Tofighi et al. [34], contacts between different agent types (patient, doctor, CCF, resident, RRT, nurse, and AHS) were calculated. Each proximity of less than 2 m is considered to be a contact. The pedestrian library of AnyLogic controls the movements of agents that perform the tasks following their routines and the contact calculator counts the contacts between the agents. Using this approach, it is possible to examine different scenarios of ICU operation and the resulting contacts between the agents.

Due to many contacts between caregivers, the probability of disease transmission among them may increase as a consequence of their exposure to infectious patients. The modeling highlights
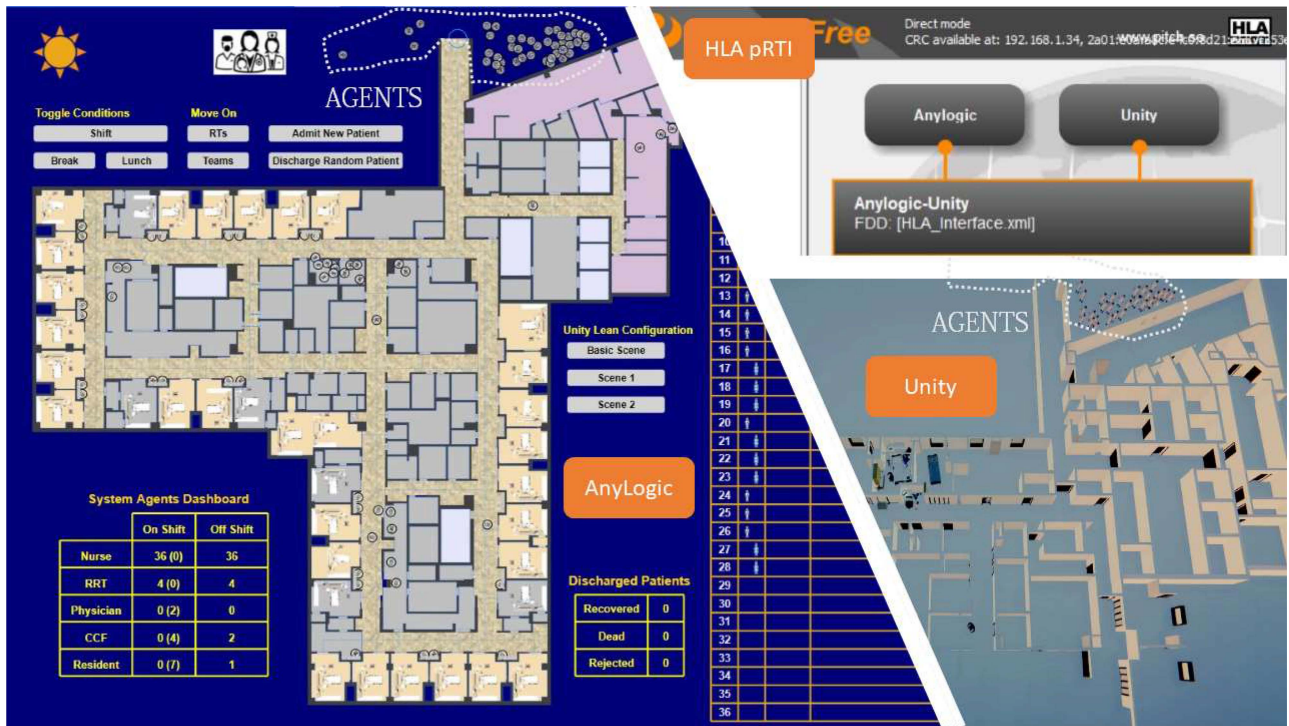
Fig. 7. AnyLogic 2-D ICU model, Unity 3-D model, and the HLA federation orchestrator.

the importance of scheduling, management of shifts, and division of work in the ICU ward in order to reduce the probability of disease transmission. This information can be used to measure the impacts of different prevention measures on the number of contacts, and as a key input for estimating disease outbreak in the ICU ward under different conditions and scenarios.

B. ICU Ward Simulation in Unity

A virtual model of the TGH ICU ward was earlier developed in Unity [49]. Unity was selected because it is a full-featured platform and provides a high-quality environment for creating interactive 3-D content that can be combined with VR hardware to develop a VR application. The 3-D environment was developed to mimic real-world arrangements of rooms, offices, equipment, space, and movements (Figs. 5 and 6), with the use of a combination of 3-D modeling software, including Autodesk Revit, Autodesk 3-Ds Max, and Blender 3-D. Autodesk Revit was used to develop the initial 3-D model of the building in a fairly efficient and accurate manner. Autodesk 3-Ds Max and Blender 3-D modeling software were used to create objects—e.g., the virtual patient, hospital furniture, and protective equipment. Fig. 5 shows an ICU room in the VR application. Asgary et al. [49] had sought to prepare a safe platform for training healthcare providers on critical ICU measures relating to COVID-19 patients, particularly on 1) donning and doffing of personal protective equipment and 2) intubation of COVID-19 patients with severe respiratory conditions.

In the current ICU simulation model in Unity, the virtual patient (see Fig. 6) has two main attributes as follows:
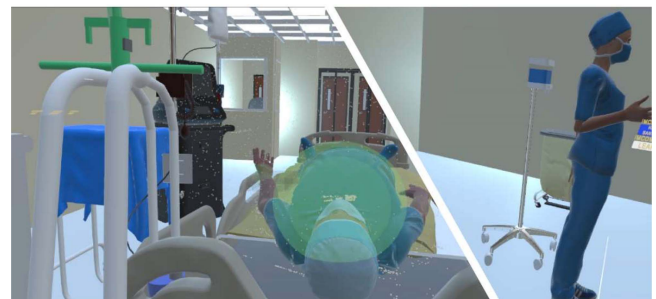

Fig. 8. Unity 3-D modeling and virus spread simulation.

1) It has been designed and modeled so that it responds to physical interactions between the patient, the healthcare providers, and the equipment attached to the patient.
2) A disease particle system has been added to visualize COVID-19 virus spread.

To address the second attribute, three particle generating objects are attached to the mouth of a virtual patient or infected person. Each particle subsystem simulates one of the three states of regular breathing, coughing, and sneezing. For regular breathing, we simulated a person's breathing every 3 s in which particles are spread within a 1-m radius with a speed of 0.3 m per second. For the coughing state, we modeled coughing with different frequencies in which particles are spread with a speed of 1 m per second and reaching up to a 2-m radius. For the sneezing case, we considered a speed of 5 to 10 m per second for particles with a possible reach of an 8-m radius. To calculate the number of particles reaching each virtual object's surface or
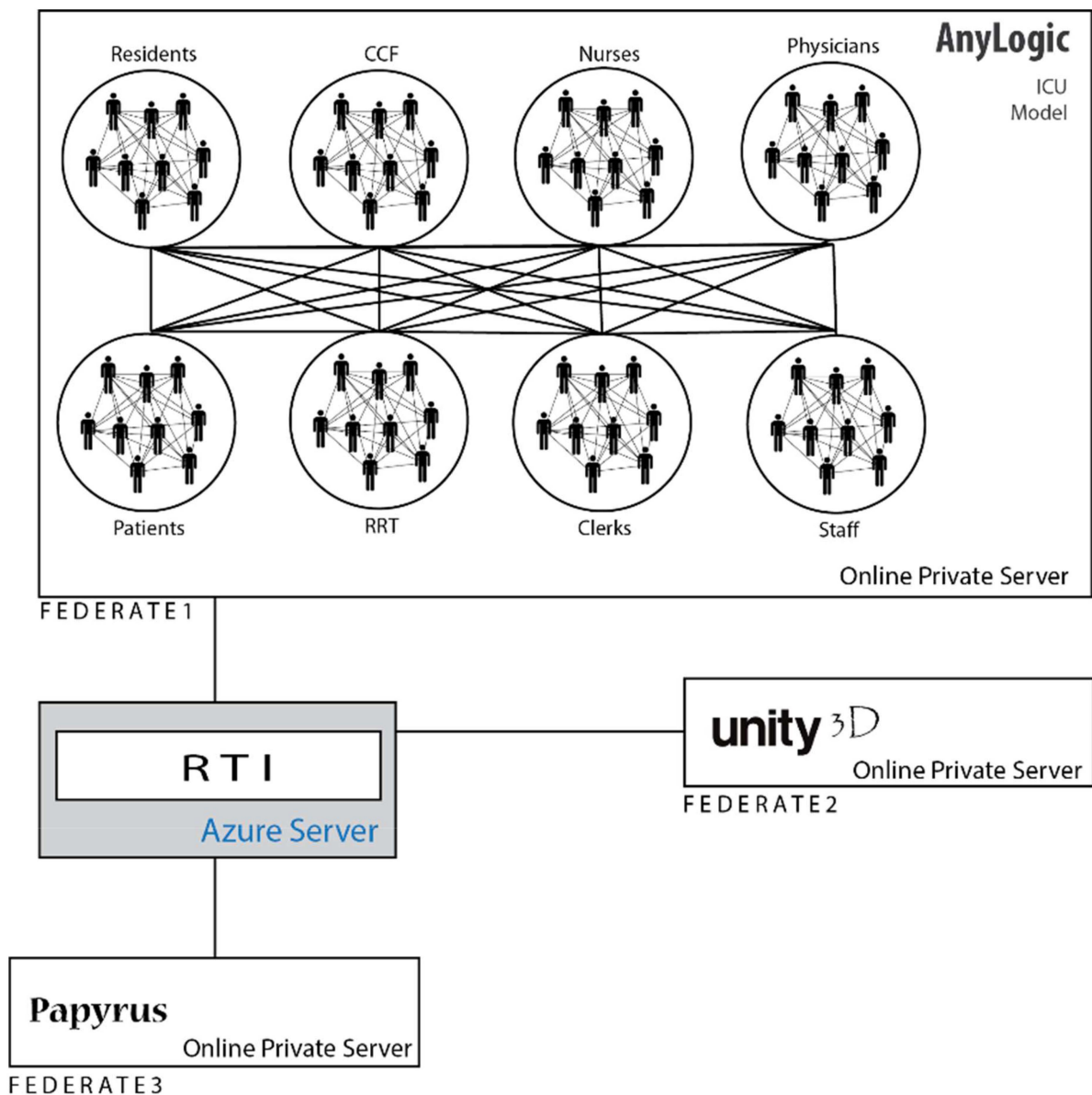
Fig. 9.   DS system architecture.

body, we created a script in Unity that is attached to each object and is able to count the number of particles colliding with the object.

C. Integrating the ICU AnyLogic Simulation and the ICU Unity Simulation Models

Fig. 7 depicts the following:
1) The AnyLogic-developed 2-D ICU model (on the left).
2) The Unity-developed 3-D ICU model (on the lower right) operating in parallel with the 2-D model.
3) The HLA federation, with AnyLogic and Unity linked as federates (on the upper right).

Small circles are used to represent agents in the 2-D model. During the simulation run, the agents' relevant data are published to the RTI, to be constructed as 3-D avatars in Unity.

Through the p/s mechanism of HLA, AnyLogic publishes the agent (nurse, physician, patient, etc.) data—ID, Type, and

Position (X Y Z)—every 0.3 s to the online RTI. Unity, configured as an HLA federate connected to the same RTI and subscribed to the data, receives the position and attributes it to the appropriate agent, based on the received agent ID and Type. Using the experiments of Jankovic [50], we visualized virus contamination in the hospital environment from an infected person to surrounding surfaces and close individuals using Unity particle systems (Fig. 8). The number of particles is sent from Unity through the publish mechanism and received by AnyLogic through the subscribe mechanism of HLA.

Papyrus is used as a master UML modeler responsible for the planned procedures orchestration of the DS system. It is installed on a private online server. AnyLogic, as well, is installed on a private online Windows 2019 server. Data are sent from AnyLogic to Unity through the HLA communication. Some information, such as the number of virus particles, is sent from

Fig. 10. Normalized average chunks of effective contacts per day, based on seven 24-h replications of the ICU simulation model.

Unity to AnyLogic through the common RTI installed on Azure (Fig. 9).

We were accordingly able to combine two powerful simulation technologies (AnyLogic and Unity) using the HLA standard to create a more complete VR solution that shows the ABM and process DES in an appealing 3-D modeling interface. Average chunks of effective contacts per day, as presented in Section IV-A-4, are observed while playing the scenario in this animated and 3-D rendered visualization. In other words, users who are experimenting with the 3-D models on VR headsets would be able to observe in 3-D the live process flow coming from the AnyLogic ABM and DES application. This can assist in training professionals on how to make decisions. Moreover, some simulations, such as the spread of COVID-19 particles, are difficult, if not impossible, to construct with DES simulators. However, using Unity's particle system, such a simulation is possible to create. By way of this DS approach, the user is able to examine the particle system results during the AnyLogic simulation runs, as well as visualize the real-time process flow and ABM simulation, with the use of the Unity 3-D model (as depicted in Fig. 8).

V. DISCUSSION OF SIMULATION RESULTS

In the ICU simulation model, the base run assumes that the ICU ward is running at less than full capacity, with 35 patients. We use a two-stage M&S approach. The first stage is simulated for 24 h to cover shifts and record agent interactions for a full day of ICU operations. By recording and summarizing the contacts through the calculation techniques explained in Najafabadi et al. [47], we extracted an effective contacts matrix (Fig. 10) that summarizes disease transmission patterns among the human agents. We have used a normalized weighted average of the length of such contacts, yielding the average frequency of the contacts between each pair of agent types, as earlier depicted in Fig. 10. The generated matrix will be symmetric almost, except for the *Patient* agent type, which is only inserted in the rows but not the columns (since all patients are assumed to be already infected and, thus, there would not be any patient-to-patient

transmission that the model should capture). A contact between each pair of agents is recorded at both ends, by each of the agents involved.

The frequencies appearing in Fig. 10 can be plugged into the following compound viral transmission probability equation to come up with an infection probability matrix that can be used in the simulation model for tracing the disease propagation:

$$\text{Probability of Viral Transmission } = 1 - (1 - p)^f$$

where p is the base probability of virus transmission in one effective contact and f is the quantified frequency of the effective contacts between a pair of agent types.

The model was developed in collaboration with the medical staff of the ICU. The generated contacts matrix was an intriguing simulation outcome for the ICU medical staff to identify more risky roles and schedules in terms of increasing viral transmission inside the unit, to more effectively plan/manage their work in order to reduce the likelihood of viral spread.

For the second stage of M&S, a platform was designed to graphically depict the disease transmission model findings for all existing human agent types in the system, in addition to the contacts matrix results of Fig. 10. Fig. 11 illustrates the outputs of the disease transmission simulation model, albeit only for three agent types (residents, nurses, and doctors) in view of space considerations. This platform shows how different types of medical staff in the ICU go through different stages of the disease. It depicts the number of exposed, asymptomatic, symptomatic, and recovered medical staff per agent type during a 45-day simulation period. The implementation of the disease transmission state chart has earlier been described in subsection IV-A-2. The simulator collects and draws the findings based on the contacts matrix and the agent-based model parameters and configuration during the simulation run.

In Fig. 11, for example, two residents were exposed to an asymptomatic agent at the start of the simulation run. The number of residents who have been exposed has increased to five by day 5. Some of the residents who are exposed become infected (symptomatic or asymptomatic). The recovered residents begin to appear on the graph on day 11 of the simulation. By looking at the output graphs for different types of medical staff agents, we can see clear distinctions on how and when different stages of the disease are developing for each type of staff. This can be used to confirm the effects of different work schedules on the risk levels associated with various medical agent types.

The major goal of this article is to show that, using the IEEE HLA standard and its associated mechanisms, it is possible to integrate diverse heterogeneous components in real time to create a powerful agent-based simulation/VR tool. This technology has expanded the simulation horizon. A discrete event and agent-based simulator now manage the simulation agents during the VR experiment, allowing medical staff to immerse themselves in a virtual world that is as near to reality as possible. This technology has been evaluated in a health-related area, but it might be adopted in other fields as well.
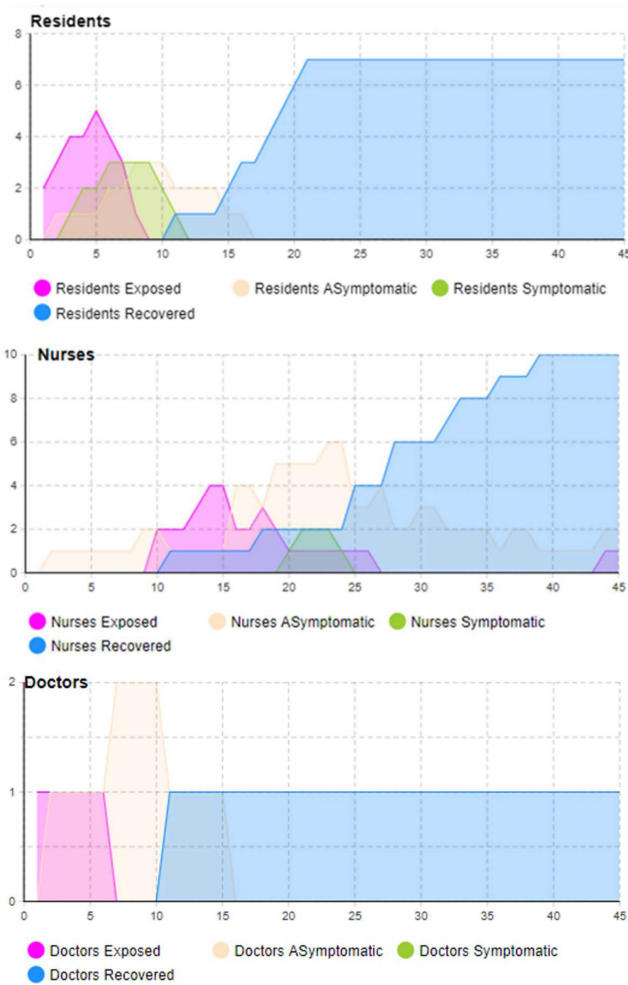
Fig. 11. Sample simulation results: Numbers of exposed/asymptomatic/symptomatic/recovered medical staff in the ICU during the 45-day simulation.

VI. CONCLUSION

In this article, we present a cloud-based DS system developed for the ICU of a hospital based in Toronto, Canada for managers to have visual models and predictive analytics that help them in finding solutions to improve the effectiveness, quality of services, and safety in highly communicable disease environments. The interoperability issues between the existing heterogeneous components installed on different operating systems and different cloud servers are tackled using the HLA IEEE DS standard. Figs. 7 and 9 illustrate the global architecture used to develop the cloud-based DS system.

The scope of the work reported in this article is the development of a cloud-based DS system in the healthcare domain, specifically in a hospital ICU ward, with the situation made more complex by the COVID-19 pandemic. The main role of the HLA standard is to solve the interoperability issues between the cloud-based UML modeler, ABM, and DES simulator, and 3-D engine system components. Papyrus is used as a general and master modeler, Unity as a visualization engine, and AnyLogic for agent-based and discrete event modeling. The developed DS system enables users to interact with the simulation

and examine/review different scenarios and their impacts. It may be used for training purposes and assessment of different operational management strategies to minimize contacts and disease transmission in the ICU ward.

The established DS system paves the way for the agent-based simulator to be integrated with a VR environment, allowing end users to examine the effectiveness of new training and management approaches and methodologies to improve ICU ward operations especially during the ongoing pandemic. To this end, the DS system will further be tested by and on several relevant end users.

A. Limitations of the Article

In the current phase of this article, the article has focused on how to use the developed DS to trace interactions among human agents, resulting in an effective contacts matrix, and accordingly track disease transmission within the ICU. The application work is currently limited to these modeling aspects.

In addition, the models reported here do not consider portable equipment, such as portable X-ray machines, mobile dialysis machines, temporary beds, and other equipment which move from room to room.

B. Future Article

We intend to expand the use of this DS in assessing a variety of operational and process management scenarios, as well as innovative methodologies and system approaches that assist in improving efficiency and process gains, as well as resolving logistical and scheduling issues. It is also worth mentioning that the DS architecture is designed to be compatible with and readily integrated (as modules and interfaces) into other models that are not necessarily health related.

In future article, portable/mobile devices could be introduced as agents that may increase the likelihood of virus transmission within the unit. Moreover, we also intend to investigate new strategies to personalize the agents' experience by adding more players inside the ICU ward. In addition to a patient, a physician, or a nurse, other relevant players for the ICU may include a nurse's aide, a transporter, or a ward cleaning staff member. Building on the actions and activities of various relevant players will allow a more effective predictive model. We also wish to investigate adding reinforcement learning capabilities.

REFERENCES

[1] P. Punnakitikashem, J. M. Rosenberger, and D. B. Behan, "Stochastic programming for nurse assignment," *Comput. Optim. Appl.*, vol. 40, no. 3, pp. 321–349, 2008.
[2] P. Punnakitikashem, J. M. Rosenberber, and D. F. Buckley-Behan, "A stochastic programming approach for integrated nurse staffing and assignment," *IIE Trans.*, vol. 45, no. 10, pp. 1059–1076, 2013.
[3] N. Gerdsri, A. Kongthon, and S. Puengrusme, "Profiling the research landscape in emerging areas using bibliometrics and text mining: A case study of biomedical engineering (BME) in Thailand," *Int. J. Innov. Technol. Manage.*, vol. 14, no. 2, 2017, Art. no. 1740011.
[4] Y. Bouanan, M. B. El Alaoui, G. Zacharewicz, and B. Vallespir, "Using DEVS and Cell-DEVS for modelling of information impact on individuals in social network," in *Proc. IFIP Int. Conf. Adv. Prod. Manage. Syst.*, 2014, pp. 409–416.

[5] M. Sbayou, G. Zacharewicz, Y. Bouanan, and B. Vallespir, "BPMN co-ordination and DEVS network architecture for healthcare organizations," *Int. J. Privacy Health Inf. Manage.*, vol. 7, no. 1, pp. 103–115, 2019.

[6] S. Barnes, B. Golden, and S. Price, "Applications of agent-based modeling and simulation to healthcare operations management," in *Handbook of Healthcare Operations Management*, B. T. Denton, Ed. New York, NY, USA: Springer, 2013, pp. 45–74.

[7] A. Erdemir et al., "Credible practice of modeling and simulation in healthcare: Ten rules from a multidisciplinary perspective," *J. Transl. Med.*, vol. 18, no. 1, pp. 1–18, 2020.

[8] J. Possik et al., "A distributed simulation approach to integrate any-logic and unity for virtual reality applications: Case of COVID-19 modelling and training in a dialysis unit," in *Proc. IEEE/ACM 25th Int. Symp. Distrib. Simul. Real Time Appl.*, 2021, pp. 1–7, doi: 10.1109/DS-RT52167.2021.9576149.

[9] P. Keskinocak and N. Savva, "A review of the healthcare-management (modeling) literature published in manufacturing & service operations management," *Manuf. Service Oper. Manage.*, vol. 22, no. 1, pp. 59–72, 2020.

[10] F. Gao and A. Sunyaev, "Context matters: A review of the determinant factors in the decision to adopt cloud computing in healthcare," *Int. J. Inf. Manage.*, vol. 48, pp. 120–138, 2019.

[11] W. Yu, Q. Liu, G. Zhao, and Y. Song, "Exploring the effects of data-driven hospital operations on operational performance from the resource orchestration theory perspective," *IEEE Trans. Eng. Manage.*, to be published, 2021.

[12] S. Bag, S. Gupta, T.-M. Choi, and A. Kumar, "Roles of innovation leadership on using big data analytics to establish resilient healthcare supply chains to combat the covid-19 pandemic: A multimethodological study," *IEEE Trans. Eng. Manage.*, to be published, 2021.

[13] S. C. Brailsford, T. Eldabi, M. Kunc, N. Mustafee, and A. F. Osorio, "Hybrid simulation modelling in operational research: A state-of-the-art review," *Eur. J. Oper. Res.*, vol. 278, no. 3, pp. 721–737, 2019.

[14] N. Mustafee and J. H. Powell, "From hybrid simulation to hybrid systems modelling," in *Proc. Winter Simul. Conf.*, 2018, pp. 1430–1439.

[15] A. O. Solis, A. Asgary, J. Nosedal-Sánchez, and B. Zaccaro, "Developing a fire response simulation test bench based on NFID. Project report submitted to the Canadian Association of fire chiefs," Mar. 2018. [Online]. Available: https://cjr.ufv.ca/developing-a-fire-response-test-bench-based-on-nfid/

[16] A. O. Solis et al., "Agent-based simulation of a fire department's response to emergency incidents: An updated model.," in *Proc. 9th Int. Defence Homeland Secur. Simul. Workshop*, Sep. 2019, pp. 39–46.

[17] C. Ruiz-Martin, G. Wainer, Y. Bouanan, G. Zacharewicz, and A. L. Paredes, "A hybrid approach to study communication in emergency plans," in *Proc. Winter Simul. Conf.*, 2016, pp. 1376–1387.

[18] P. Bocciarelli, A. D'Ambrogio, A. Falcone, A. Garro, and A. Giglio, "A model-driven approach to enable the simulation of complex systems on distributed architectures," *Simulation*, vol. 95, no. 12, pp. 1185–1211, Feb. 2019, doi: 10.1177/0037549719829828.

[19] A.-M. Tamas, L. Kiss, and M.-M. Coman, "Distributed simulation–a connection vector among Romanian joint national training center and simulation training centers during large-scale training exercises," *J. Defense Resour. Manage.*, vol. 8, no. 1, pp. 77–82, 2017.

[20] G. S. Blair, M. Paolucci, P. Grace, and N. Georgantas, "Interoperability in complex distributed systems," in *Proc. Int. Sch. Formal Methods Des. Comput., Commun. Softw. Syst.*, 2011, pp. 1–26.

[21] A. Garro and A. Falcone, "On the integration of HLA and FMI for supporting interoperability and reusability in distributed simulation," in *Proc. Symp. Theory Model. Simul.: DEVS Integr. M&S Symp.*, 2015, pp. 9–16.

[22] S. Gorecki, J. Possik, G. Zacharewicz, Y. Ducq, and N. Perry, "A multi-component distributed framework for smart production system modeling and simulation," *Sustainability*, vol. 12, no. 17, 2020, Art. no. 6969, doi: 10.3390/su12176969.

[23] M. Lees, B. Logan, and G. Theodoropoulos, "Distributed simulation of agent-based systems with HLA," *ACM Trans. Model. Comput. Simul.*, vol. 17, no. 3, pp. 11–es, Jul. 2007, doi: 10.1145/1243991.1243992.

[24] J. Possik, A. D'Ambrogio, G. Zacharewicz, A. Amrani, and B. Vallespir, "A BPMN/HLA-based methodology for collaborative distributed DES," in *Proc. IEEE 28th Int. Conf. Enabling Technol.: Infrastruct. Collaborative Enterprises*, 2019, pp. 118–123, doi: 10.1109/WETICE.2019.00033.

[25] F. Kuhl, R. Weatherly, and J. Dahmann, *Creating Computer Simulation Systems: An Introduction to the High Level Architecture*. Hoboken, NJ, USA: Prentice Hall, 1999.

[26] B. Möller, M. Karlsson, R. Herzog, and D. Wood, "Security in simulation—New Authorization opportunities in HLA 4," in *Proc. Virtual Simul. Innov. Workshop*, 2021, pp. 1–10, Paper 2021_SIW_Paper_13. [Online]. Available: https://www.sisostds.org/DigitalLibrary.aspx?EntryId=52238

[27] S. Gorecki, Y. Bouanan, G. Zacharewicz, J. Ribault, and N. Perry, "Integrating HLA-based distributed simulation for management science and BPMN," *IFAC-Papers Online*, vol. 51, no. 11, pp. 655–660, 2018.

[28] J. Possik, "Contribution to a methodology and a co-simulation framework assessing the impact of lean on manufacturing performance," Ph.D. dissertation, IMS Lab, Univ. de Bordeaux, Bordeaux, France, 2019.

[29] G. Zacharewicz, C. Frydman, and N. Giambiasi, "G-DEVS/HLA environment for distributed simulations of workflows," *Simulation*, vol. 84, no. 5, pp. 197–213, 2008.

[30] B. P. Zeigler, S. B. Hall, and H. S. Sarjoughian, "Exploiting HLA and DEVS to promote interoperability and reuse in Lockheed's corporate environment," *Simulation*, vol. 73, no. 5, pp. 288–295, 1999.

[31] C. M. Macal, "Everything you need to know about agent-based modelling and simulation," *J. Simul.*, vol. 10, no. 2, pp. 144–156, May 2016, doi: 10.1057/jos.2016.7.

[32] C. M. Macal and M. J. North, "Tutorial on agent-based modelling and simulation," *J. Simul.*, vol. 4, no. 3, pp. 151–162, Sep. 2010, doi: 10.1057/jos.2010.3.

[33] N. A. Choudhary, M. Ramkumar, T. Schoenherr, and N. P. Rana, "Assessing supply chain resilience during the pandemic using network analysis," *IEEE Trans. Eng. Manage.*, to be published, 2021.

[34] M. Tofighi et al., "Modelling COVID-19 transmission in a hemodialysis centre using simulation generated contacts matrices," *PLoS One*, vol. 16, no. 11, 2021, Art. no. e0259970.

[35] I. Grigoryev, *AnyLogic 8 in Three Days*, 5th ed., 2018. [Online]. Available: https://www.anylogic.com/upload/al-in-3-days/anylogic-in-3-days.pdf

[36] É. Malbos, R. Oppenheimer, and C. Lançon, Se Libérer Des Troubles Anxieux Par La Réalité Virtuelle, Collection Eyrolles Pratique, Editions Eyrolles, 2017.

[37] C. J. Turner, W. Hutabarat, J. Oyekan, and A. Tiwari, "Discrete event simulation and virtual reality use in industry: New opportunities and future trends," *IEEE Trans. Human-Mach. Syst.*, vol. 46, no. 6, pp. 882–894, Jun. 2016.

[38] X. Huang, M. White, and M. Burry, "Design globally, immerse locally—A synthetic design approach by integrating agent based modelling with virtual reality," in *Proc. 23rd Int. Conf. Assoc. Comput.-Aided Architectural Des. Res. Asia*, 2018, pp. 473–482.

[39] AnyLogic, 2020. [Online]. Available: https://www.anylogic.com/

[40] PrtI, Pitch Technologies, 2021. [Online]. Available: www.pitch.se

[41] J. Frijters, IKVM, 2014. [Online]. Available: http://www.ikvm.net/

[42] *Microsoft Azure*, Microsoft, 2021. [Online]. Available: https://azure.microsoft.com/en-ca/

[43] S. Gorecki, J. Ribault, G. Zacharewicz, Y. Ducq, and N. Perry, "Risk management and distributed simulation in papyrus tool for decision making in industrial context," *Comput. Ind. Eng.*, vol. 137, Nov. 2019, Art. no. 106039, doi: 10.1016/j.cie.2019.106039.

[44] M. Y. Li and J. S. Muldowney, "Global stability for the SEIR model in epidemiology," *Math. Biosci.*, vol. 125, no. 2, pp. 155–164, Feb. 1995, doi: 10.1016/0025-5564(95)92756-5.

[45] S. He, Y. Peng, and K. Sun, "SEIR modeling of the COVID-19 and its dynamics," *Nonlinear Dyn.*, vol. 101, no. 3, pp. 1667–1680, Aug. 2020, doi: 10.1007/s11071-020-05743-y.

[46] A. Asgary, M. G. Cojocaru, M. M. Najafabadi, and J. Wu, "Simulating preventative testing of SARS-CoV-2 in schools: Policy implications," *BMC Public Health*, vol. 21, no. 1, Dec. 2021, Art. no. 125, doi: 10.1186/s12889-020-10153-1.

[47] M. M. Najafabadi, A. Asgary, M. Tofighi, and G. Tofighi, "Generating simulation-based contacts matrices for disease transmission modelling at special settings," 2021, *arxiv: 210110224*.

[48] "Interim guidance on developing a covid-19 case investigation & contact tracing plan (close contact)," CDC, 2020. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/php/contact-tracing/contact-tracing-plan/appendix.html#contact

[49] A. Asgary et al., "Use of virtual reality for training on procedures in an intensive care unit during the COVID-19 pandemic," in *Proc. 10th Int. Workshop Innov. Simul. Healthcare*, Sep. 2021, pp. 76–83.

[50] L. Jankovic, "Experiments with self-organised simulation of movement of infectious aerosols in buildings," *Sustainability*, vol. 12, no. 12, 2020, Art. no. 5204.