Proceedings of the 2015 Winter Simulation Conference
L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, eds.

OBJECT ORIENTED FRAMEWORK FOR HEALTHCARE SIMULATION

Akshay Venkitasubramanian
Stephen D Roberts

Edward P Fitts Department of Industrial and Systems Engineering
North Carolina State University
400 Daniels Hall
Raleigh, NC 27695, USA

Jeffrey A Joines

Department of Textile Engineering, Chemistry and Science
North Carolina State University
1000 Main Campus Drive
Raleigh, NC 27695, USA

ABSTRACT

Healthcare is a highly interconnected dynamic environment where multiple combinations of individuals and teams come together to serve patients. Because of the interconnections in the healthcare system, a multi-facility, flexible simulation modeling methodology is desirable, where the modeling boundaries are flexible enough to capture the complex interactions between service centers. This paper presents a flexible framework for multi-facility simulation using an object-oriented simulation paradigm specifically designed for health care services. To test the framework's capabilities, a multi-facility simulation model of the patient flow at a University Student Healthcare Clinic (SHC) was implemented using this proposed framework and standard out-of-the box methods. Based on this modeling and implementation experience, the authors reflect on the utility of a healthcare oriented framework versus standard out-of-the-box simulation tools for healthcare simulation projects.

1 DISCRETE EVENT SIMULATION (DES) AND HEALTH CARE OPERATIONS

Virtually every segment of the health care delivery operation has employed Discrete-Event Simulation (DES), owing partly to the growth of computing power and the availability of user friendly software. The extensive literature has been surveyed periodically – see England and Roberts (1978), Jacobson, Jun, and Swisher (1999), and Fone et al. (2003). A consistent theme seen is that despite the growth of simulation modeling in the literature, there remains limited evidence of implementation. Hall, Jacobson, and Swisher (2006) noted the lack of investigations into complex integrated multi-facility systems or those that examined inter-departmental relationships. Günal and Pidd (2010) concluded that the reuse of models for other scenarios or hospital is very limited. Katsaliaki and Mustafee (2011) also note that models that are flexible and reusable to solve a variety of scenarios are needed as well as modeling paradigms to facilitate development of multi-facility environments.

Some commercial simulation language developers have attempted to address the health care delivery field. MedModel was developed in the late 1990s by Promodel Corporation. It extended the functionality of their general simulation language, ProModel, to match lab results to patients, preempting resources, staff schedules and subroutines. The MedModel package also came with an extensive 2D healthcare symbol library (reference). The Flexsim HC^{TM} is the healthcare extension of Flexsim, a general simulation package. The Flexsim HC^{M} provides 3D custom objects to represent patient queuing, processing and resources with an in-depth focus on providing statistics in a graphically interactive way. Nonetheless, the commercial packages have a fixed structure . They do not facilitate extensibility of classes and limit reusability of objects. The user of these packages do not have the ability to customize objects to meet their needs if the objects provided cannot fully or easily model the user's situation. This limits the applicability of these languages to address complex healthcare problems.

One modeling paradigm that may accommodate the reusable and flexible multi-facility need for simulation modeling is the Object-Oriented Simulation (OOS) approach. The fundamental concepts in OOS, such as classes, composition, inheritance, polymorphism, and run-time binding, were first presented by Dahl and Nygarrd (1966). Dessouky and Roberts (1998), present a comprehensive review of the OOS approach. Some elaboration of OOS is found in Joines and Roberts (1998) which adds the concepts of frames and frameworks. The parallel execution of object-oriented simulations is discussed in Baezner and Lomow (1991). Finally, a perspective on the event, process, and object "world views" of simulation modeling is given by Pegden (2010).

One of the earliest examples of OOS in the healthcare environment was used to model a hospital's bed resource utilization using SMALLTALK (Birch and Cook 1991). The authors concluded that the visual user interface helped in gaining a higher user acceptance. Moreover, the administration did a periodic review of the simulation model and the changes were implemented with ease given the OOS nature of the language. For a more detailed systematic review of healthcare simulation literature as well as OOS, we ask that the reader to refer to Akshay (2012).

Using the OOS approach, software objects (also called agents) are developed that provide a wide range of behaviors specially adapted to multi-facility environments. These objects are specified for the particular systems being simulated, allowing objects to be reused from application to application. Furthermore, if the objects are defined appropriately then they can be integrated into a multi-facility system. The challenge is to design these objects so that they can be applied to develop effective and efficient health care simulations.

2 THE HEALTHCARE TOOLKIT

In this research, a comprehensive object-oriented simulation framework for healthcare is presented that overcomes the flaws of commercial packages. This toolkit provides a means for easy creation of complex integrated multi-facility simulations based on the object-oriented paradigm. In this section, we provide the software architect's plan for this development, which is approximately independent of the implementation software (design and implementation cannot be completely independent). This framework provides a conceptual basis for healthcare that can be employed by an object-oriented simulation language. Obviously, such a basis is not fully compressive of all health care delivery processes, but does create a groundwork that can be further extended.

Healthcare, as an industry, lacks standardization in service organization and delivery due to its heavy dependence on people at both sides of the service table. This lack of standardization has led hospitals and other health care organizations to adapt processes to suit their organization and to best cater to their patient demographics. It also partly explains the redundancy of numerous simulation models that describe the same operation, such as the large number of emergency department simulations (Günal and Pidd 2010, Hall, Jacobson, and Swisher 2006). Nonetheless, this "uniqueness" poses a significant challenge to healthcare simulation modelers as they need to customize standard simulation constructs to reproduce these complex behaviors. Most often, this affects the cost-effectiveness of the simulation model. Thus, good healthcare simulation modeling software should have basic structures to model the operations, and at the same time, have built-in flexibility to incorporate the modeler's requirements with relative ease.

From a simulation software design point of view, we view healthcare operations as composed of three basic object classes (i.e., Stationary, Mobile, and Agent objects). Stationary objects represent fixed hospital structures that provide essential hospital services such as "Registration Desks," "Patient Beds," "Laboratory Equipment," "X-ray machines," etc. Mobile objects are movable objects that move between these hospital structures in order to facilitate services such as stretchers, wheelchairs, crash-carts, nurses, doctors, etc. Finally, Agents are representative objects that undergo transactions or receive services from stationary and mobile objects. Agents represent actual patients, paperwork, specimens, etc. These three broad object classes encapsulate the majority of healthcare operations. A high-level object hierarchy is presented in Figure 1 to clarify the concepts defined in this section.

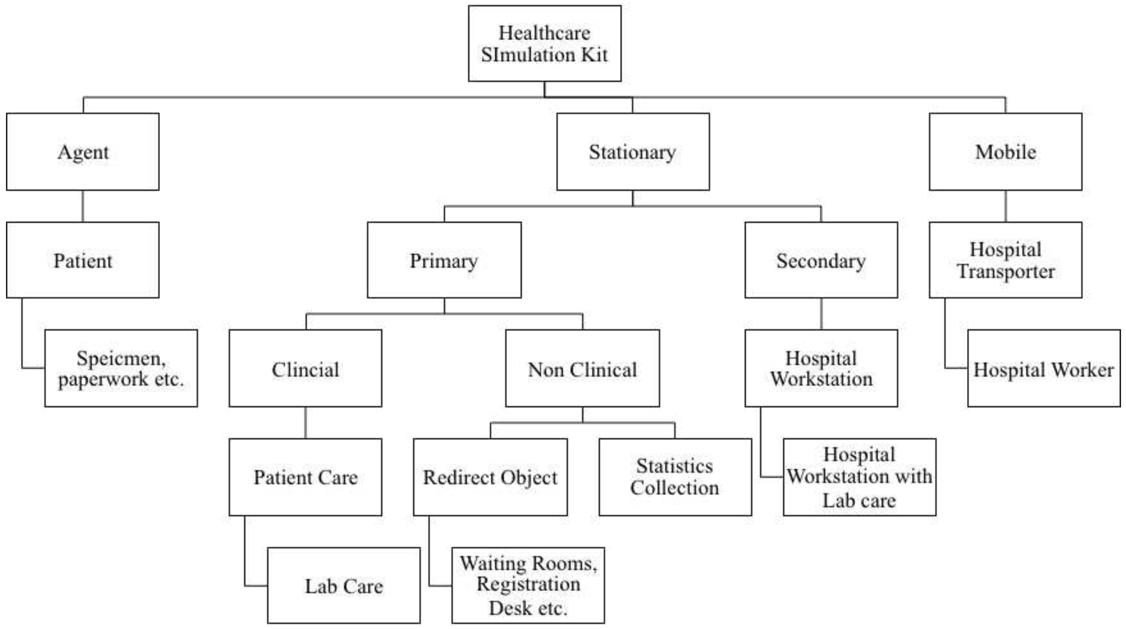

Figure 1: Healthcare operations simulation hierarchy.

Hospital operations will be used to frame the remaining discussion. However, you should recognize that the basic concepts and approaches can apply to a number of health care related operations including rehabilitation, nursing homes, clinical labs, outpatient surgery, clinics, doctor's offices, etc. There is an implicit assumption that the agent is an active participant within the health care delivery process, so this operations structure is less appealing for problems where the patient is stationary or less active, such as operating rooms or recovery areas.

2.1 Agent Objects

Agent objects represent a class of objects that receive service in hospitals. Typically, these are patients or some associated form, such as a specimen or paperwork. Every healthcare system's operational efficacy is also strongly influenced by patient behavior, which should be replicated by the logic embedded in this class. This class of objects is central to the design of an effective healthcare simulation toolkit. In addition, there should be provisions to store patient preferences as they directly affect patient behavior such as a patient's requirement for a wheelchair. Another factor that must be given attention is a provision for the modeler to describe the stages that each patient has undergone (triage, treatment, x-ray visit etc.). This information provides a meaningful breakdown of the system's statistical performance.

For every patient who enters a healthcare service, there may be paperwork generated. These paperwork entities represent patients and most of the processes in the hospital await paperwork to begin service and generate paperwork at the end of service. Secondly, a majority of medical service providers heavily depend on secondary medical services such as lab testing, radiology, stress tests, etc. to assist with treatment diagnosis and action plans, where there is usually a specimen from the patient. The specimen undergoes processing in a different environment, where a patient sometimes is not required to be present. For post-processing, the specimen gets converted into paperwork, which should be then matched with the right patient. In both cases, the agents are replicas of patients, and thus can be developed by extending the **Patient Entity** class. The **Specimen Entity** class is a logical replica of the patient and accepts services in place of the patient. This class must have the intelligence to identify its patient (parent) to whom it belongs, so that they can be matched further downstream. Synchronization of patients and their related information is a characteristic prominent in healthcare, but less so in other industries.

2.2 Stationary Objects

Agents from the previous section require a network in which to flow and interact. Stationary objects are the largest class category and their domain can be divided in two categories: Primary and Secondary. Primary stationary objects are those that interact directly with the patient. These include hospital beds, registration desk, waiting room, etc. On the other hand, secondary stationary objects deal with a reference to the patient such as specimen in laboratory, paperwork in the hospital records room, a prescription in a pharmacy and so forth.

2.2.1 Primary Stationary Objects

The primary stationary object design needs to encompass general features of a hospital that interfaces directly with the patient (interacts with an agent we call the **Patient Entity**). These objects can be clinical or non-clinical in nature, whereby clinical objects serve a medical purpose and non-clinical objects support the clinical structure. To clarify, by clinical we refer to hospital facilities such as treatment beds or x-rays, and we term this group of objects **Care** objects. Non-clinical objects refer to facilities such as waiting areas, registration desks, or billing desks.

2.2.1.1 Redirect Object

A typical hospital or clinic is comprised of many waiting rooms, registration desks, help desks, and nurses' stations, among many others. These are the places where more than one patient can queue up to be redirected for clinical or non-clinical activities. Hence, we take advantage of this functionality by designing a **Redirect Object**. In a nutshell, these objects keep hospital structures glued together by providing a means to integrate facilities. Furthermore, it is advantageous to design a framework that decouples patient queuing from clinical activities in order to facilitate a realistic simulation of the healthcare system. From a design perspective, the **Redirect Object** models clinical facilities that facilitate patient waiting as well as routing. The **Redirect Object** must also support logical expressions to model routing based on patient factors (reneging and balking behavior), clinical policies (FIFO, most severe patient, longest waiting time, etc.) as well as collecting statistics related to efficiency and efficacy of patient routing (congestion factor, active load, etc.)

Based on the above specified design, the developer can easily extend it to model a variety of administrative structures such as registration desks, billing desks, etc. Although these non-clinical structures are functionally different, their base structure requires only a slight modification of the **Redirect Object** class.

2.2.1.2 Patient Care Class

The simplest clinical object construct that addresses patient modeling concerns is a patient processing object (**Patient care**). This object is designed to be a simple processing station with no input or output buffer. The reason for the adoption of this design is to force patients to wait at designated locations, which are modeled by non-clinical objects, specifically at **Redirect Object** classes, thereby enhancing the fidelity of the simulation model.

2.2.1.3 Lab Care Class

Modern medicine heavily relies on diagnostic services like laboratory testing and diagnostic imaging as well as many other forms of testing for treatment. These objects interact with patients and support secondary care functions, which can be broken down into three different sub-processes: specimen collection, specimen processing, and result dispatching. At the **Lab care** level, we need to design processes to handle the interactions at specimen collection and results dispatch. The specimen processing aspect is handled by a hospital workstation object class, discussed later. The specimen collection process and results dispatching processes are inherently dependent on clinical policies, patient preference and test requirements. Furthermore, the specimen processing aspect needs to be modeled at the patient level to capture patient-patient variability. This approach represents a natural way of thinking for the modeler, since the decision is associated with a patient and facilitates development of customized logic to capture decision modeling.

2.2.1.4 Statistics Collection Class

The primary objective of a simulation is to understand the system, often from the patient's perspective. In all hospital operational simulations, the primary factor of interest has often been to improve some aspect of the patient care process such as the response time for treatment, reducing turnover time from the lab, and so on. The **Statistics Collector** class should be designed to collect these types of predefined statistics automatically from the **Patient Entity** as they exit the system. We expect the collection of absolute statistics as well as computing relative statistics.

2.2.2 Secondary objects

Secondary objects are those hospital functions that support patient care without directly interacting with the actual patient, such as a laboratory department which processes patient specimens. These facilities are important to the healthcare simulation toolkit because modern medicine heavily relies on these services and these operations are often overlooked by simulation modelers (Bodtker et al. 1993). In this section, we deal specifically with the design of the specimen processing systems.

2.2.2.1 Hospital Workstation

After a specimen is collected from the patient using the **Lab care** object, the specimens undergo processing at the laboratory. The laboratory processes are generally standardized, which makes them ideal to encapsulate into an object. In essence, any lab process can be broken down into three stages: setup, processing, and post processing. The setup stage represents the processes that transform the specimen into an acceptable form to be processed by the machine. The processing stage encapsulates the logic for analyzing the specimen and performing the required tests. The post processing stage captures the result preparation logic which could range from preparing paperwork to processing an image or even just packaging the processed results. At every stage, the specimen can be processed individually or in a batch, Batching and un-batching logic should be defined at each stage as well as modular object design to handle the resource deadlock problem. The **Hospital Workstation** object should also support changeover processing logic to account for operational variability between specimens as well as inventory/material management to support inventory management issue modeling.

2.2.2.2 Hospital Workstation with Lab care

Not all types of patients' secondary visits are specimen controlled; some require the patient to be present for the initial specimen collection. For example, a patient with a fractured limb is directed to an X-ray imaging service, and this requires that the patient visit a service center to initiate the service. The **HWS_Labcare** object class models these classes of systems by combining the logic from **Hospital Workstation** and **Lab Care** classes. The resulting dispatching logic is similar to that presented in the **Hospital Workstation** section with two key differentiators: outsourcing and machine failures.

In some services, the post-processing stage is outsourced. Outsourcing can also be modeled using the same object by allowing the release of the resource after **Patient Entity** exits the processing stage. In these types of facilities, there are some exceptions to the dispatching of results logic. There are some cases where patient reports are instantly handed over to them after the tests are performed, which fall under Wait at Current Object redirection logic. To incorporate this combination, we need to append additional result batching logic to this class.

Finally, the major addition to this object is the concept of "*Failures"*. Most of the machines modeled by **HWS_Labcare** class are known to be composed of a number of electro-mechanical parts whose reliability is often in question. Failures of these facilities are known to be a major hassle for managers, and play an important role in facility operations. Failures can also conceptually represent machine downtimes, such as when the operator is on a break or off-shift. Thus, failures may be grouped into categories such as: Calendar Time, Processing Time, Processing Count, and Event Count. The implementation method behind failure would be similar. A monitor is needed to internally police when the event specifications depend on a particular failure type.

2.3 Mobile Objects

Mobile objects can be categorized into two main categories: Transporters and Workers. Transporter objects only interact with agents and assist in moving them from one point to another. Workers assist the patient with hospital services such as nurses, doctors, etc. In this section, we develop design specifications for both a **Hospital Transporter** and a **Hospital Worker**.

From a design perspective, a transporter is a movable resource, which a patient seizes to reach their destination. Thereby, we can leverage the base design of the standard resource and add an efficient routing logic specific to a hospital environment. The routing algorithm can be broken down into two modules: navigation and transportation.

The navigation aspect of the algorithm is responsible for helping the transporter move within the hospital network. The navigation algorithm is responsible for determining if the transporter has arrived at it destination. There are three ways that a destination can be determined:

1. It can be the default place for the transporter to be stored in a hospital,
2. It can be the location of a patient who has requested it (if free), or
3. It can be the destination the patient needs to reach (if occupied).

Once the transporter arrives at its destination, we first determine if it is the default place of storage. If not, then it has to either pick up (if free) or drop off (if occupied). This picking up, dropping off, or transportation action may require the assistance of a hospital employee (for example, a patient in a wheelchair). If this is the case, then we need to embed additional resource seize/release logic within the transporter class.

The **Hospital Worker** class can be viewed as a special case hospital transporter class. In essence, a nurse in a hospital moves around the hospital to serve patients as well as help patients navigate the hospital. However, nurses work on shifts and may have to perform patient rounds. These can be easily attached to the existing **Hospital Transporter** class. We can implement standardized scheduling constructs for facilitating worker shifts. As for routing, we can develop fixed routing which can hold a set of destinations which fall under the nurse's purview. This process can be overridden if necessary with a higher priority task to account for emergency cases.

2.4 Implementation

In order to implement the aforementioned healthcare framework, the SIMIO language (Pegden and Sturrock, 2010), an object oriented simulation language, is chosen. A system-centric view over an agent-centric framework is adopted, since the former has a superior run time performance in complex simulation models. We named our healthcare framework SIMIO-HC. The SIMIO-HC base classes can be grouped into four groups which are: Fixed, Entity, Node, and Transporter. It should be noted that the base classes are analogous to standard SIMIO base classes. Figure 2 shows the relationship between the SIMIO and the SIMIO-HC classes with the standard SIMIO objects in grey, which illustrates that our objects are extensions/specializations of the standard objects.

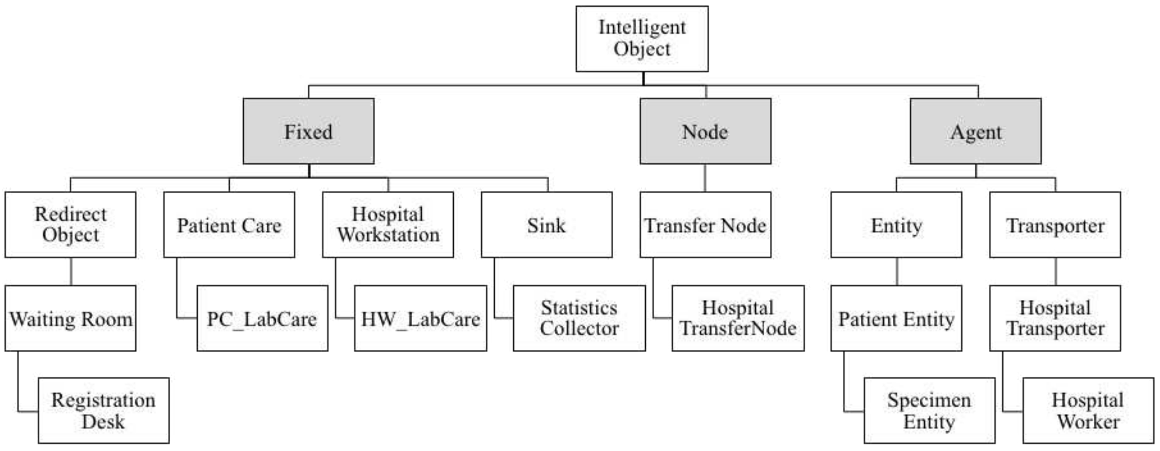

Figure 2: SIMIO-HC class hierarchy.

Each class in the SIMIO-HC hierarchy corresponds to the previous discussions. The implementation documentation of the SIMIO implementation are found in Akshay, (2012). SIMIO HC is an OOS, and users can utilize SIMIO language/constructs to augment the health care objects to create more specialized health care operations.

3 APPLICATION TO PATIENT FLOW IMPROVEMENT AT STUDENT HEALTH SERVICES CLINIC

To illustrate the application of the framework, we present a case study where we model the patient flow of the North Carolina State Student Health Services (SHS). The SHS provides care for a student population of approximately 30,000 people. With a student population this large, even with their best efforts, SHS is unable to meet the expected service level, thereby causing long waiting times at the clinics. This concern, in turn, leads to patients staying beyond normal work hours, resulting in frequent overtime for personnel. Hence, we were charged with developing a model that could improve the patient flow and reduce resource utilization without compromising the services rendered by SHS.

The SHS is composed of five medical clinics. Each clinic has two physicians and one nurse The health center runs four clinics to serve patients via appointments through their online portal. The last clinic serves walk-in patients who require medical care but were not able to make an appointment.

The appointments are given out in slots of 20 minutes each; however, these 20 minute intervals are often not enough to provide the necessary care to patients. The walk-in clinic was established to deal with emergency medical cases and thus serves patients depending on severity rather than the order of arrival. As a result, it is often used by patients who were not able to secure appointments or were late for their appointments.

Currently, each clinic operates individually. Patients wait in a queue outside of each clinic. They wait for a nurse to call them for initial triage and then wait in one of the six clinic rooms until a physician is free to treat them. Once treated, a patient goes to the lab for tests, the pharmacy to fill prescriptions, exits the system, or follows any combination. If a patient is directed to the lab, they may be required to visit the clinic after the appointment.

3.1 Model Setup

Two types of SHS models were built. The first model was built using the SIMIO Standard library exclusively. We shall refer to this model as the Standard Model. The second model used SIMIO-HC objects along with the SIMIO standard library objects, and thus will be referred to as the Simio-HC Model. This approach helped the team to qualitatively assess the efficacy of the SIMIO-HC toolkit.

In both models, we used the “building blocks” approach where the simulation model is composed of two or more facility specific models. This approach allows the simulation project to be broken down into smaller units of individual facilities. Each of the individual facilities are modeled, verified and validated independently and then combined in their final stages. Hence, we can leverage the expertise in facility-specific modeling of the simulation community to develop complex simulation models. Furthermore, both models were built to mimic the SHS clinic in structure. This approach was taken in order to ensure that the models are easy to understand, especially by the SHS stakeholders.

In order to implement the Standard Model, we had to modify the standard server object’s input and output buffer capacity to zero to ensure no entity will be allowed to wait at a clinic except when undergoing service. The waiting room was implemented with a combination of Server and Transfer node, where the latter had to be modified to implement FIFO under stochastic condition. The same waiting room and server logic was replicated to model radiology and laboratory section. The laboratory processing required another server and required the creation of a reference entity to model processing operations, which is similar to the Hospital Workstation logic. On the other hand, the SIMIO-HC model was implemented directly “out of the box”. The respective objects were directly dragged from the toolkit library and their properties were modified to satisfy model requirements.

3.2 Verification and Validation

The simulation model was continuously verified throughout its creation as part of the generic debugging process. The model was monitored using the model trace and watch functions in SIMIO to ensure that the entity flow from object to object worked as intended in the correct sequence, and without error.

Two types of model validation were implemented: visual and statistical. The model was animated and the patient flow was visually validated. For statistical validation, total time in system for the model entity from the simulation models was compared with the average total time computed from the raw data in the time study. One-hundred replications of the model were run to get accurate estimates of the total time in system with 95% confidence. The simulation models recorded an average time in system for all entities at 49.4 ± 1.43 minutes (SIMIO-HC Model) and 48.6 ± 1.13 minutes (Standard Model) which fell within the observed confidence interval for average time in system 49.7 ±0.74 minutes.

4 DISCUSSION AND CONCLUSIONS

The vision behind the development of SIMIO-HC toolkit is to develop a framework which can accelerate simulation model development without compromising on modeling detail. In other words, the SIMIO-HC toolkit should be easy to use and at the same time, be flexible enough to model a wide range of healthcare operations. This hypothesis was qualitatively assessed by developing a model both with and without the toolkit. Overall, our experiences with the Standard Model lacked the detail to instill faith in stakeholders since it did not directly communicate within the healthcare language. In other words, the outcome metrics (such as utilization categories) or input property labels (such as capacity) did not adhere to standard healthcare terminology. This communication gap adds a layer of complexity that can be avoided by using healthcare specific simulation framework such as the SIMIO-HC toolkit. Such an advantage would be attributed also to the commercial health care simulation languages versus the general simulation languages.

From a development point of view, the increase in modeling complexity directly translates to an increase in the time and cost to develop the simulation model. The increase in cost and time makes the simulation option unattractive as the returns (insights and experimentation) are not cost -effective. In our experience, the Standard Model took longer to build given the additional customizations that needed to be done to the standard SIMIO library. Furthermore, we had to add additional output states for generating custom statistics that were required to drive experimentation. Overall, the Standard Model development required a strong understanding of the SIMIO tool and healthcare functional knowledge. On the other hand, the SIMIO-HC Toolkit development process accounted both for the tool and healthcare domain. This simplified the modeling of the SHS system for the team without compromising modeling fidelity, thereby improving the modeling experience for modelers and stakeholders. For example, we had to heavily rely on statistical validation for the Standard Model as compared to the SIMIO-HC model, where both statistical and visual validation could be implemented. Visual validation accelerated the model validation process and reduced the barrier for stakeholder buy-in.

Only eight percent of the surveyed literature indicated that the application of simulations to real world healthcare problems (Eldabi 2009) resulted in implementation, while the application of Complex Integrated Multi-Facility Simulation Models (CIMS) in Healthcare comprises only three percent of the reviewed literature (Günal and Pidd 2010, Jacobson, Jun, and Swisher 1999). Günal and Pidd (2010), and Hall, Jacobson, and Swisher (2006), have already established a positive relationship between CIMS models and implementation of a simulation model. However, there is no facilitative framework for CIMS, which can be addressed by using a “truly” object oriented simulation toolkit. A “truly” object oriented simulation toolkit will also bridge the disconnect between simulation practice and healthcare as the simulation models are built around objects rather than programming logic The SIMIO-HC toolkit is a first step in this direction with the aim of providing a framework for an object oriented toolkit for healthcare simulations with support for building CIMS models.

The SIMIO-HC toolkit provides basic classes that can be extended to customize for the modeler’s need and accelerate the development process. Furthermore, the toolkit is designed with usability and modeling accuracy being central issues as these factors make simulation models easy to comprehend for stakeholders who are often unfamiliar with simulation modeling. Finally, simulation models must be easy to maintain and allow for future customization, which is once again addressed by the OOS framework. Overall, the SIMIO-HC toolkit may accelerate the development of healthcare simulation models without compromising on modeling fidelity, thereby making simulation modeling a more cost effective and attractive option for healthcare problems.

Although the toolkit presented here has many of the essential base classes, there exist ample opportunities to refine the framework. The toolkit currently does not have costing variables and costing elements and these could be added based on the activity-based costing available in SIMIO. Secondly, the open framework of SIMIO and SIMIO-HC allows for users to take apart the classes and re-configure them. We hope this helps in further refining the toolkit and that it will also lead to the creation of domain-specific specializations of SIMIO-HC. In addition, hospitals are filled with resources such as crash carts which are not fixed at a location. These resources can be aptly named as **Movable Resources** and a future class should represent these resources. Finally, the animation provided in SIMIO is three-dimensional in nature, which provides realistic views of the simulation model. Prior research has demonstrated that animation positively influences model acceptance by stakeholders, which is critical to successful implementation. Further interactive hospital graphical elements should be included as SIMIO upgrades its graphical capabilities in order to engage the modeler and stakeholders in active model development.

REFERENCES

Akshay, V. 2012. “Object-Oriented Framework for Healthcare Simulation.”, Masters Thesis, North Carolina State University

Baezner, B., and G. Lomow, 1991. “A Tutorial Introduction to Object-Oriented Simulation and Sim++.” In Proceedings of the 1991 Winter Simulation Conference, edited by B. L. Nelson, W. D. Kelton, and G. M. Clark, 157-163. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Bevan, G., and S. Palmer. 2003. “Systematic Review of the Use and Value of Computer Simulation Modelling in Population Health and Health Care Delivery.” Journal of Public Health Medicine 25:325-335

Birch, G., and S. Cook. "Discrete Event Simulation in Smalltalk/V Windows.", In 19991 IEE Colloquium on. IET, 1-3.

Bodtker, K., W. Godolphin, and L. Wilson. 1993. “Simulation Modelling to Assist Operational Management and Planning in Clinical Laboratories.” Simulation 60:247-255

O. Dahl and K. Nygaard, 1966. “SIMULA: an ALGOL-based Simulation Language.” Communication of the ACM 9: 671-678

Dessouky, Y. M., and C. A. Roberts. 1998. “An Overview of Object-Oriented Simulation.” Simulation 70:359-368

Eldabi, T. 2009. “Implementation Issues of Modeling Healthcare Problems: Misconceptions and lessons.”, In Proceedings of the 2009 Winter Simulation Conference, edited by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, 1831-1839. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

England, W., and S. Roberts. 1978. “Applications of Computer Simulation in Health Care.” In Proceedings of the 1978 Winter simulation Conference, edited by H.J. Highland; N.R. Nielson; L.G. Hull, 665-676. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Fone, D., S. Hollinghurst, M. Temple, A. Round, N. Lester, A. Weightman, K. Roberts, and E. Coyle. 2003. "Systematic Review of the Use and Value of Computer Simulation Modelling in Population Health and Health Care Delivery." Journal of Public Health 25(4): 325-335.

Günal, M. M., and M. Pidd. 2007. “Interconnected DES Models of Emergency, Outpatient, and Inpatient Departments of a Hospital.” In Proceedings of the 2007 Winter Simulation Conference, edited by S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton. 1461-1466. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Günal, M. M., and M. Pidd. 2010. “Discrete Event Simulation for Performance Modelling in Health Care: A Review of the Literature.” Journal of Simulation 4:42-51

Hall, S., S. Jacobson, and J. R. Swisher. 2006. “Discrete Event Simulation of Healthcare Systems.” In Patient flow: reducing delay in healthcare delivery, edited by R. W.Hall. Springer pp. 211-252

Jacobson, S. H., J. B. Jun, and J. R. Swisher. 1999. “Application of Discrete-Event Simulation in Health Care Clinics: A Survey.” Journal of Operations Research Society 50:109-123.

Katsaliaki, K. and N. Mustafee. 2011. “Applications of Simulation within the Healthcare Context.” Journal of Operations Research Society 62:1431-1451

Joines, J. A., and S. D. Roberts. 1998. “Object Oriented Simulation.”, In Handbook of Simulation - Principles, Methodology, Advances, Applications, and Practice, edited by J. Banks, John Wiley & Sons, 397-426, 605-627

Pegden, C. D. 2010. “Advanced Tutorial: Overview of Simulation World Views.” In Proceedings of the 2010 Winter Simulation Conference, edited by B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, 210-215. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Pegden, C. D., and D. T. Sturrock, 2010. “Introduction to Simio.” In Proceedings of the 2010 Winter Simulation Conference, edited by B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, 1-10. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGROPHIES

AKSHAY VENKITASUBRAMANIAN is a product manager at Remedy Partners, a CMMI awardee convener for Bundled Payments Collaborative Initiative (BPCI) and manages development clinical care coordination products. Akshay received his M.S.I.E from Edward P Fitts Department of Industrial and Systems Engineering at North Carolina State and B.S. in Biomedical Engineering from Cochin University of Science and Technology. His research interests are focused on application simulation and mathematical modeling to health systems. His e-mail address is vsa.akshay@gmail.com.

STEPHEN D. ROBERTS is the A. Doug Allison Distinguished Professor in the Edward P. Fitts Department of Industrial and Systems Engineering at North Carolina State University. Professor Roberts received his Ph.D., M.S.I.E., and B.S.I.E. from the School of Industrial Engineering at Purdue University. His research interests include simulation and health systems engineering. His email address is roberts@ncsu.edu.

JEFFERY A. JOINES is an Associate Professor and the Associate Department Head of Undergraduate Studies in the Textile Engineering, Chemistry, and Science Department at NC State. He received a B.S. in Electrical Engineering and B.S. in Industrial Engineering in 1990, a M.S in Industrial Engineering in 1993, and Ph.D. in Industrial Engineering in 1996 all from NC State University. He received the 1997 Pritsker Doctoral Dissertation Award for the best dissertation from the Institute of Industrial Engineers. His expertise is in supply chain optimization utilizing computer simulation and optimization, and he has published numerous papers and given dozens of international conference presentations. He was the Co-Proceedings Editor for the 2000 Winter Simulation Conference and the Program Chair for the 2005 Winter Simulation Conference, and is currently the IEEE representative on the WSC Board. He teaches undergraduate and graduate classes in computer information systems, computer based modeling in Excel and VBA, simulation and six-sigma. He was awarded the 2006 NC State University Outstanding Teaching Award and the 2012 Alumni Distinguished Undergraduate Professor.