HCCM - A CONTROL WORLD VIEW FOR HEALTH CARE DISCRETE EVENT SIMUALTION

Nikolaus Furian
Michael O'Sullivan
Cameron Walker
Department of Engineering Science
University of Auckland
Auckland, New Zealand
E-mail: nfur003@aucklanduni.ac.nz

Siegfried Vössner
Institute of Engineering and Business Informatics
University of Technology Graz
8010 Graz, Austria
E-mail: voessner@tugraz.at

KEYWORDS
Health Care, Discrete Event Simulation, Activity Scanning, Simulation Control

ABSTRACT

The classical world-views of discrete event simulation (DES) are event scheduling, activity scanning and process interaction. A fourth approach, the three-phase method, extends activity scanning and is often regarded as another world-view. These world-views provide the theoretical framework for applying DES in practice. However, in health care simulation, practitioners often face modeling challenges where the concepts and methodologies described by these world-views are not able to reflect either the dynamics or the entity flow of the system being modelled. This leads to individualized approaches and solutions that do not build on a unified and standardized theoretical basis. In this paper we present an extension to the activity scanning world-view based on the needs of the health care sector that uses hierarchical control structures as a more general, flexible and powerful tool to define health care DES models. To demonstrate the strength and potential of the approach two ongoing simulation studies are briefly outlined and the benefits of using the new world-view for these models is discussed.

INTRODUCTION AND MOTIVATION

In DES the underlying system is changed at discrete points in time. How these changes are modeled, controlled and triggered is dependent on the sub-domain of discrete event simulation, also known as the *world-view*, being utilized. The classical world-views are: event scheduling; activity scanning; and process interaction. However, the three phase method, which is an extension to the activity scanning approach, is also often referred to as a world-view. Much work has been done to formalize, evaluate, compare and transform the different world-views, as in (Balci 1988; Birta and Arbez 2007), or more recently (Overstreet and Nance 2004).

Recently the need for a uniform description of discrete event models has been identified (Robinson 2006; Balci et al 2008; Robinson et al. 2010). However, a uniform descriptive language, that is generally accepted and used, requires that the underlying simulation theory can be used to model the majority of simulation projects. Unfortunately, although most published models in the health care sector, are based on one of the classical world-views, these models often include customized features that are not included in that world-view, to fit the requirements of the specific system being modeled, as for example (Günal and Pidd 2011) who generate multiple "mini-doctors" out of a doctor to imitate multi-tasking.

In this paper we will outline some of the major drawbacks of the activity scanning and three phase method world-views for health care simulation followed by the description of concepts that attempt to resolve these shortcomings and lead to more generality and flexibility within activity based DES. The paper is structured as follows. In the following section we discuss the three phase method world-view and the ABCmod framework that developed from it. In the next section we revisit the concept of activities, followed by a definition of the hierarchical control mechanism. Section *Time Advancement* briefly describes time advancement for the proposed method. The paper concludes with two case studies from health care, suggestions for further research and final remarks.

THE THREE PHASE METHOD AND ABCmod

In the original three phase method, see for example (Pidd 1995), the only concept of control is that of conditional activities in the entity flow within a model. Based on this method, (Birta and Arbez 2007) introduced the most structured and deliberate modeling theory and language for DES, the ABCmod framework. Besides conditional activities, they identify queues as core elements of any activity based DES to control the entity flow.

Although, queues and conditional activities are able to reflect less complex, more rigid systems, such as those that often occur in manufacturing problems, they often fail to represent health care systems with complicated, dynamic dispatch policies, resource flows and entity relationships. The need for more flexible control

Proceedings 28th European Conference on Modelling and Simulation ©ECMS Flaminio Squazzoni, Fabio Baronio, Claudia Archetti, Marco Castellani (Editors)
ISBN: 978-0-9564944-8-1 / ISBN: 978-0-9564944-9-8 (CD)

structures in DES has been addressed previously by (Pratt et al. 1991), who outlined the importance of separating control and informational elements of a model. For health care modeling, (Hay et al. 2006) pointed out that sophisticated dispatch systems and the skill set of staff members drive the process rather than patients in simple queues. A more advanced approach has been proposed by (Lim et al. 2013), where entities can assign tasks to other entities. Definitions of structured control mechanisms for process and job oriented simulation has been introduced by (Raunak et al. 2009), (Mes and Bruens 2012) and (Robinson et al. 2006).

The standard approach based on conditional activities and queues can be briefly summarized: as soon as a condition evaluates to true an entity is chosen from a queue (e.g. first-in-first-out or priority based) and an activity is triggered. This approach requires the independence of queues from each other and a relatively rigid mapping of resources to queues. However, health care scenarios often consist of: more complicated dispatch methods; resources that regularly change their role to consumers and vice versa; time-dependent priorities; and skill-level dependent job allocations. Furthermore, one often faces a high degree of interruptions and concurrent tasks. Therefore, dispatch policies have to select from a pool of future and uncompleted jobs, taking into account a very heterogeneous set of factors including: time related priorities; task related priorities; degree of completion; future assigned tasks; skill levels; and individual preferences.

When modelling using classical queuing with conditional activities, practitioners quickly encounter the limitations of this approach and create customized workarounds. The simulation control concept presented in this paper replaces conditional activities and queues by a hierarchical control tree. This tree consists of nodes that represent control units with activities at the leaves. In this approach more general rule sets replace dispatch via queues and conditions, thus providing greater flexibility when modelling.

Furthermore, the hierarchical control world-view enables the integration of optimization within simulation. The classical approach to the integration of simulation and optimization uses a simulation model to evaluate an objective function of an optimization model. Thereby, the simulation model and its implementation act as a black box. Hence, both systems, the simulation and optimization, are encapsulated and only interact via the exchange of parameters and objective values. In addition to this standard approach, the hierarchical control world-view, presented in this paper, allows optimization techniques to be elegantly embedded within a simulation model to control the simulation flow itself. Although, optimization has been embedded within simulations in previous applied research, a uniform theoretical basis for integrating optimization techniques to control entity flow is still missing in literature.

REVISITING ACTIVITIES

Entities and their behavioral artifacts are two of the main elements of any DES. Behavioral artifacts represent the flow of entities and their relationships and interactions with each other. Usually, two types of behaviors are defined, events (or actions) and activities. *Events* cause instant changes to the simulation model's state, for example the arrival of an entity. The classical definition of *activities* is that they stretch over a certain period of time, consist of at least two events: the start; and end event; and represent a purposeful task.

In the classical world-view literature, events and activities are classified as scheduled, conditional or sequential activities depending on their trigger mechanism.

Scheduled events and activities occur at designated simulation times, independent of the simulation model's state. Further, they represent the mechanism to advance time during simulation. Activity scanning and the more advanced three phase method both scan through all *conditional* events and activities each time the simulation clock progresses and triggers an event or activity if a condition evaluates to true. *Sequential* events and activities are triggered immediately after the termination of a preceding behavioral artifact. In particular, activities are denoted as scheduled, sequential or conditional if their start events are of the corresponding type.

In the hierarchical control world-view scheduled and sequential behavior is handled in the standard way, but a single class for conditional behavior is not considered sufficient to enable general, flexible simulation models, particularly with respect to simulation control and advanced dispatch.

To demonstrate the shortcomings of the standard classification of activities and to illustrate the proposed hierarchical control world-view we will use a simple patient transport simulation. Patients located at a ward require escorting to a diagnostic facility. We assume that these requests occur randomly according to some kind of stochastic process. They stay in the ward until they are guided to an empty diagnostic site, in the diagnostic facility, by an orderly (we assume that there is only one orderly in the model), an activity called escort-to. Note that there are only a limited number of diagnostic sites available in the facility. In the diagnostic site patients receive some sort of diagnostic assessment and are subsequently escorted back (called escort-back) once the orderly is available, i.e., escorted by the orderly from the diagnostic site to the ward. After being escorted back, the patient stays in the ward until she leaves or needs another service. Furthermore, the orderly moves by herself from the diagnostic facility to the ward if no patients are ready for being escorted back, there are patients waiting in the ward for transport and at least one diagnostic site is available. On the other hand, the orderly moves by itself from the ward to the diagnostic facility if no patients are waiting for transport and at least one patient is currently diagnosed or has finished diagnostic assessment.

According to the standard classification of activities, escort-to, escort-back and the orderly moving by herself whether from the ward to the diagnostic facility or vice versa are conditional activities. However, there is a significant difference in the trigger mechanisms of these activities.

First, both escorting (to and back from the diagnostic facility) activities are *requested* by the patient's behavior. Although, the activities are handled and triggered by the control tree, they are motivated by the patient's behavioral path. Hence, we refer to them as *requested* activities.

On the other hand, the orderly moving by herself has different motivations and trigger mechanisms. The behavioral path of the orderly does not trigger an empty move. These moves are determined by the control policies of the model and triggered by the hierarchical control tree, hence are called *controlled* activities.

However, there can be a significant difference in the trigger mechanisms of controlled activities. When a patient needs a diagnostic assessment she requests being escorted to the diagnostic facility. When the hierarchical control tree assigns the orderly to this request, it will trigger the orderly to move by herself to the ward if necessary. Similarly, the control tree will trigger an orderly to move to the diagnostic facility when it assigns the orderly to a request for escorting a patient back after the patient's assessment is finished. However, if no patient has filed an escort-request and at least one patient is being assessed, but has not completed this assessment (in the diagnostic facility) then there is no request to assign the orderly to and, hence, trigger the orderly to move by itself. This movement is instead triggered by the hierarchical control tree because it should reduce the waiting time of the patient.

Hence a controlled activity can be triggered:

1) in response to a dispatched activity request; or
2) to improve the system's performance, e.g.,
   a. by anticipating future requests; or
   b. by considering where undispatched requests are located.

We distinguish between requested behavior (i.e., requested activities and events) and controlled behavior (i.e., activities and events that are triggered by the hierarchical control tree), but make the modeling distinction between controlled activities that occur in response to a activity request and controlled activities that occur as part of the control tree's overall plan, see figure 1.

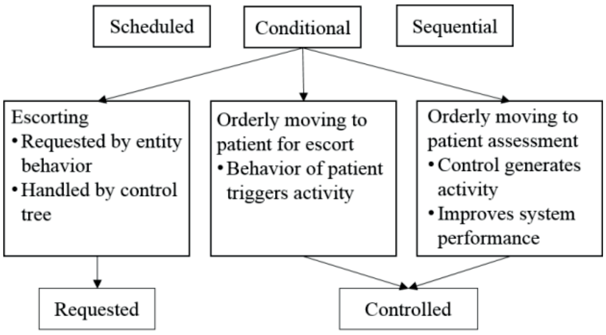

Figure 1: Extended Activity Classification

Note that requests for an activity to take place are themselves elements of the simulation model. As a consequence they may have attributes, for example the time when the request was generated, who generated it and/or an associated priority.

The definition of providing and consuming roles is part of many concepts for DES, for example in the framework of (Birta and Arbez 2007). Although their definition fits the manufacturing environment reasonably well, it is too rigid and not applicable to health care DES. In the hierarchical control world-view, any roles for entities concerning their state in terms of consuming or providing become irrelevant. For example, a less experienced surgeon assisting in a complicated surgery, as part of her educational program, cannot be unambiguously defined as either a provider (resource) or consumer (entity). On one hand, she assist during the activity and therefore actively contributes to the surgery (i.e., she is a provider). On the other hand, she benefits from the teacher-student relationship and is gaining knowledge from the more experienced surgeon leading the operation (i.e., she is a consumer). In the hierarchical control world-view, the pooling of activity requests combines with the hierarchical control tree to replace entity queues and renders the distinction between providers and consumers unnecessary.

HIERARCHICAL SIMULATION CONTROL

In the previous section one main element of the new hierarchical control world-view was introduced. Conditional activities were re-classified depending on how they were triggered in the system. In this section the other main element of the hierarchical control world-view is defined. The hierarchical control tree is introduced and request handling is described. Systems modeled using the hierarchical control world-view are hereafter referred to as Hierarchical Control Conceptual Models (HCCMs).

General Concept

One could describe discrete event models in a very informal and colloquial way as "a model that consists of a set of entities and set of tasks that have to be completed". Furthermore, behaviors and rules define the sequence of activities and the assignment of entities to activities respectively. While manufacturing systems tend to have more rigid structures, health care models often fit the informal description quite well. Patients and staff members request activities of various forms including: treatments; assessments; teaching; organizational work; rounds; meetings; and many more. These requests are hard to assign to queues as many activities require multiple resources, and resources perform multiple activities.

The hierarchical control world-view proposed in this paper follows a different approach. Instead of separating activities into different types and handling them in corresponding queues, activity requests that are associated with the same organizational area, or unit, are pooled together in Requested Activities and Events Lists (RAELs). Additionally, for each organization area, a control unit is defined to handle the corresponding activity requests. These organizational areas can be seen as sub-models of the entire system. The use of a model/sub-model structure has already been identified as a substantial requirement for model reuse and the design of integrated models, see (Zeigler 1987). We propose linking sub-models together into a hierarchical tree structure with control units at all nodes except for the leaves which are requested or controlled activities, as shown in figure 2. It is important that the control tree and requested activities are designed in an unambiguous way. In particular, the same requested activity cannot be the child of more than one control unit. Thus, activity requests are non-ambiguously handled by unique control units.

Control units manage entities, manage requested and controlled activities and handle activity requests. Further, they consist of a set of rules that determine the conditional behavior of the model, and a set of delegates that represent the communication between control units.

The depth of the designed tree is obviously a modeling choice, which has to be made under great care. Too much granularity leads to unnecessarily complex models that are cumbersome to deal with. Whereas too few control structures also can lead to rising complexity in conditions for dispatch causing interactions to get more difficult to model.

From a more technical point of view control units should also provide interfaces to obtain controlled elements, including entities, current activities and requests. Due to space limitations, a detailed, technical definition of all elements is beyond the scope of this paper and is left for future work.

In the following subsections the two main elements of a control unit's definition, rules and delegates, are explained in more detail.

Rules

Rules are one core element of control units. They replace the conditions of activities in a more structured and centralized form, and can be divided in five different categories:

- **Assessment**: What can be dispatched and/or done?
- **Dispatching**: What should be dispatched and/or done?
- **Control**: What should be done given dispatching decisions and/or the current state of the system?
- **Replacement**: What does not need to be done any more?
- **Custom Rules**

Assessment rules do not differ from common conditions in a centralized form. They only assess which activity requests from the RAEL could be dispatched and/or performed given the current state of the system, particularly in terms of available entities, and put these requests in a list of *possible* requests, referred to as RAEL (i.e., the RAEL holds only requests that can possibly be triggered in the current state of the model).

Dispatch rules decide which of the requests in the RAEL to dispatch and/or do next. In many cases they represent simple queues, but can also model various complex dispatch policies and resource allocation scenarios.

Control rules trigger behavior that is either the result of dispatch decisions (e.g., the orderly moving to perform escorting) or that are designed to improve system performance (e.g., the orderly moving in anticipation of a patient completing diagnostic assessment).

Activity replacement rules determine under which circumstances requests are removed from the RAEL. This could happen for various reasons. For example, if waiting times exceed certain limits entities might leave the model (aka *balking*), or, when an activity has been deferred (e.g., a low priority meeting), the request may have expired (e.g., the time scheduled for the meeting has passed).

Custom rules can be added to model any control functionality not sufficiently represented by the other rule categories.

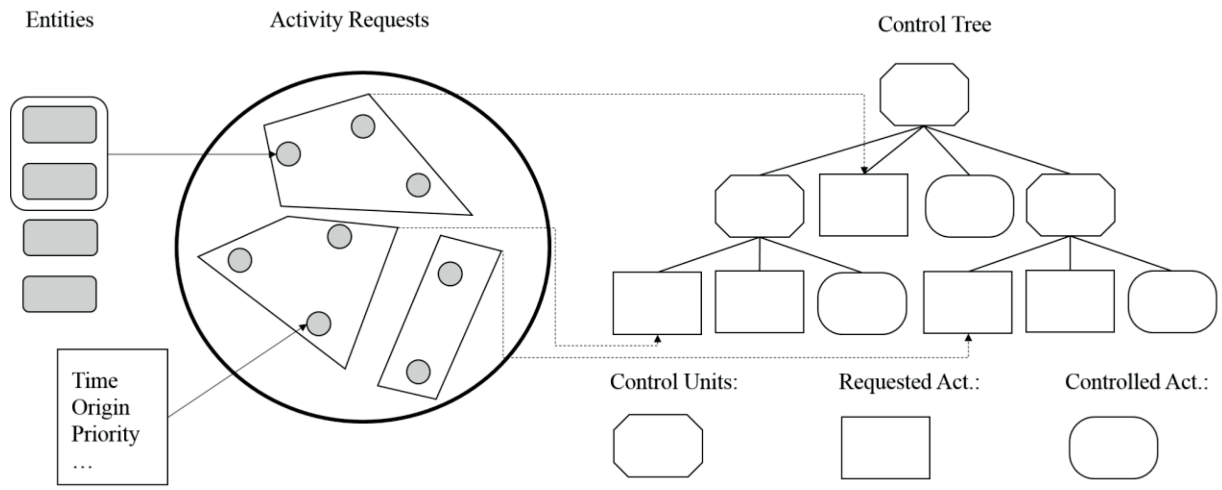

Figure 2: Concept of Hierarchical Control World View

It is important that rules are executed in the correct order. First, rules are considered in a top to bottom approach, meaning that rules of parent control units are checked first and can also differ from rules of child control units. Second, within one control unit the rule sets have to be applied in their particular order: assessment; dispatch; control; and replacement. This will be discussed in more detail in section *Time Advancement*.

Delegates

While rule sets define how control units operate and manage their entities, delegates represent the communication between control units. This messaging should strictly follow the tree structure of the control view. Thereby, bottom-up and top-down messaging is possible. Typical scenarios that require the formulation of delegates could be the report of special circumstances, for example, the occurrence of an emergency; overworked entities; the request for additional entities; or the moving of activity requests up and down the tree structure.

Basically, any kind of custom delegates can be defined. However, it has to be ensured that the receiving control unit is equipped with a corresponding rule set (receiving delegates) to handle it properly.

TIME ADVANCEMENT

In this section an algorithm for time advancement and rule execution is presented. It is based on the classical three phase method, which itself is an extension of the activity scanning world-view. Analogous to the standard three phase method, time advancement is driven by scheduled behavior. Scheduled events, either standalone or the activity start, are held in a list sorted with respect to occurrence time, referred to as the Scheduled Event List (SEL).

Each time a scheduled event is scheduled to start, the simulation clock advances and the different rule sets are used to trigger controlled behaviors. This is done in a top-down manner according to the hierarchical control tree. Thereby, the rule sets are considered in their particular order: assessment; dispatch; control; and replacement. If a requested or controlled event (standalone or activity start) has been triggered the algorithm steps back to the assessment rule set. If no action could be launched the algorithm advances to delegates. Delegates are sent both ways, bottom-up and top-down. Due to the hierarchical tree structures first delegates are sent bottom-up followed by the top-down messaging.

This process is repeated until no behavior was triggered and no delegate was sent or received. Only then the next scheduled event is considered, assuming no stopping criteria has been met. Figure 3 shows the basic principle of the time management and activity control algorithm.

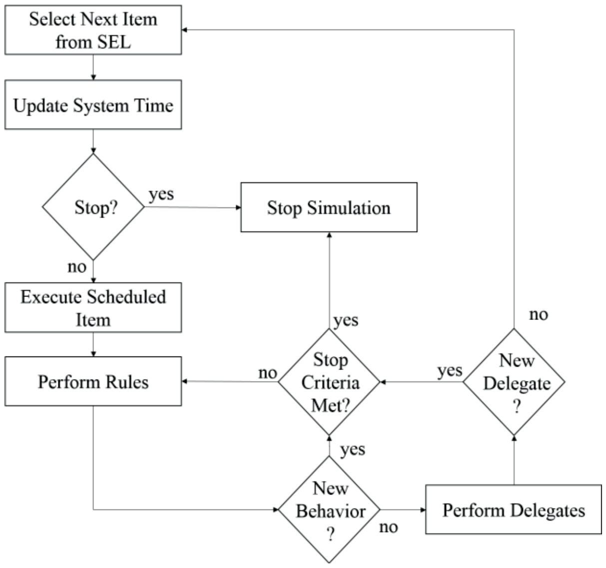

Figure 3: Time Management Algorithm

CASE STUDIES

To highlight the strengths of the proposed method two case studies are briefly discussed. For both detailed HCCM models have been created, including entity definition, formulation of event routines and all rule sets of control units. However, due to the limited space only the main characteristics of the case studies' models are outlined and the benefits of the hierarchical control world-view for these models are discussed.

Patient Transport Simulation

In an on-going simulation project at the University of Auckland, the transit of patients at Auckland City Hospital (Auckland, New Zealand) has been modeled. Patients are transported between wards and treatment facilities. These transits are performed by orderlies and, in some cases, additional nurses if required by the patient's condition. Each patient has a record of scheduled treatments he must undergo and additional emergency transports. The objective of the model is to assign orderlies (and nurses if required) to the requested transits between wards/treatment facilities in an efficient way. The request for transit is launched 40 minutes prior to the actual treatment appointment.

Using the standard queuing based discrete event simulation approach, patients that require a transport would be held in queues, either one global queue hosting all patients, or separate queues for each location in the model. As soon as resources (orderlies and possibly nurses) become available patients would be picked from the queues according to a selection routine.

However, if no other transit queued, after a completed transport orderlies return to their home base and nurses return to their home ward to wait for further assignments to transits. Hence, upon the occurrence of a transit request, usually the required resources are not present at the patient's location. This requires empty moves, i.e. controlled behavior, of both orderlies and nurses. Although the empty moves could be included in the actual transport activity, two major limitations would remain. First, the pre-emption of an empty movement with respect to the occurrence of a higher priority task, e.g. emergency transport, would require the re-insertion of the patient in the request queue at the same position the previously occupied. Second, and more important, selection of patients in queues (waiting for transit) is only triggered upon the availability of resources. However, more sophisticated dispatching routines may also take into account orderlies and nurses that are currently performing a transport. This certainly stretches the boundaries of queuing-based discrete event simulation.

The different rule sets of HCCM models presented in this paper overcome these difficulties. During assessment, jobs that can be dispatched are identified. Dispatching rules assign tasks to orderlies, independently of their availability and the control rule set triggers empty movements towards a patients and initiates actual transports. Replacement and custom rules were not necessary for this particular model.

The main focus of the project is to investigate different dispatch policies in order to identify an optimal dispatch method. The target of any dispatch routine for orderlies is that patients arrive between 15-0 minutes before the scheduled treatment, with penalties for late arrivals.

The first dispatch routine reflects the current real world strategy and assigns the nearest available orderly (and nurse, if necessary) for a new job. The second policy attempts to estimate which orderly could be at the patient's position earliest, including the time required for finishing other jobs first. The third dispatch routine really demonstrates one of the strengths of HCCM models: the dispatch rule set is “replaced”, or “represented”, by a Mixed Integer Linear Program (MILP) created from a snapshot of the system's state. Each snapshot leads to an undirected graph, where nodes represent jobs and edges represent the completion of a job plus the empty move to the location of the next job. Based on that graph a multi-travelling salesmen problem with time windows is formulated as an MILP. A solver is then used to solve the problem and the optimal solution dictates the next dispatch decision(s). Obviously optimization methods, such as MILP solvers or heuristics, have been used before to control simulation models. But the hierarchical control world-view provides a theoretical foundation to do so, which, to the knowledge of the authors, has not existed so far.

Cytology Lab Simulation

The Cytology lab study is currently under progress at the University of Auckland in cooperation with staff at LabPLUS laboratory in Auckland, New Zealand. In general, LabPLUS laboratory performs cytology analysis on different samples sent by surrounding health care facilities. These samples are analyzed by pathologists employed by the lab. In addition to sample analysis, pathologists perform a heterogeneous set of other tasks, with varying priorities, both in the lab and located in other facilities. Most of these tasks are scheduled and their date and location are known in advance. However, pathologists also participate in “call-outs”, which could be best described as emergency calls from surrounding hospitals. As it is the nature of emergencies they occur randomly and their duration is not known. Other tasks include: meetings, both in and out of the lab with varying priorities and policies for attendance; preparation for meetings; teaching; clinic visits; and organizational work.

One of the main challenges during the modeling process was the large number of interrupted activities. Pathologists constantly have to stop their current, incomplete activity to perform a higher priority task. As a consequence, while assigning jobs to pathologists, any dispatch routine has to take into account: all pending tasks; all incomplete tasks of pathologists; future scheduled activities (e.g., meetings or clinical visits); waiting times; remaining time windows for task completion (e.g., preparation time for meetings); rostering and shift data; skill levels of pathologists; blocked pathologists for certain activities (e.g., second opinions on samples); and of course the priorities of the tasks themselves. The highly heterogeneous set of activities and their dependencies was difficult to model in commercial software tools. However, a HCCM model could be elegantly stated, including the model description and the definition of dispatch routines that address all the challenges of the system.

It is not possible to outline all challenges overcome during the modeling process in detail here. However, some features that highlight limitations of the standard queuing approach and emphasize the strengths of HCCM models are illustrated in the following.

First, one feature of the model was the inclusion of training sessions for junior pathologists. Each junior pathologist has to participate in a certain number of one on one training sessions provided by a senior pathologist each month/week, depending on her skill level. Hence, with respect to the standard queuing paradigm the junior pathologists would be resources and entities at the same time. This makes independent evaluation and selection routines of queues impossible. By pooling all activity requests, regardless of being generated by a sample, pathologist or other entity more general selection functions can be formulated.

This leads to the second distinct feature of the HCCM model. As requests for different categories of tasks, with varying participants and necessary resources, are pooled together, more general dispatching functions can be defined. All possible tasks (assessment rules) are weighted according to their type, request time, degree of completion and priority. Dispatching functions then match activities and their weights with appropriate pathologists to decide what happens next. Thereby, future dispatched tasks (e.g. meetings) are also taken into account. The advantage is that jobs of different kinds are handled in one structure, and dependencies of queues that would be cumbersome to model, become irrelevant.

An additional challenge was the modeling of some kinds of meeting. While external meetings are modeled in a straight forward way, internal staff meetings are trickier to handle. Staff meetings are scheduled meetings that require all members of the lab to participate, unless they are currently engaged in, or booked in the near future for, a higher priority activity. However, the meeting only takes place when a minimum number of available pathologists (quorum) is exceeded. Otherwise, the meeting is postponed until enough pathologists become free or the latest meeting end time has been reached. Further, it is possible that a pathologist leaves the meeting earlier, e.g. due to the occurrence of an emergency, and re-joins it later. Obviously, this is a very complex situation, and very difficult to model with standard queuing and conditional activity based conceptual modeling frameworks, in an elegant way. However, the formulation of requests leads to an elegant formulation of the problem. The meeting entity generates an initial request resulting in the assignment. At the same time requests for joining are filed by non-available pathologists. Future, dispatching decisions with respect to corresponding rules allow pathologists to join once they are free. Replacement rules remove requests after the latest start-time of the meeting.

FUTURE RESEARCH

HCCM provides generality, flexibility and modularization that lead to a variety of research opportunities and application possibilities. One such example is the coupling of optimization techniques with the hierarchical control tree to simulate the potential of optimal behavior. However, the two most important goals for the near future are to define a uniform description language for HCCM and provide a software solution to implement HCCM models without great effort. Both tasks are currently under progress at the University of Auckland in cooperation with commercial simulation software providers.

CONCLUSION

In this paper we have argued that the standard theory on DES simulation has some major shortcomings for health care simulation. As a result commercial software tools and applied simulation studies often use individually customized models to fit the requirements of real world applications. This leads to significant barriers for interchangeability and re-usability of models, as well as a scientific discussion on health care DES methods. The hierarchical control world-view, and hence HCCM models, make one step towards resolving the limitations of activity-based DES. Based on an extended classification for conditional activities, a new simulation control mechanism is introduced that replaces the restricted queuing paradigm. Thereby, a much higher degree of generality and flexibility in modeling is realized. Further, the modular structure of the approach allows the integration of optimization techniques to control the simulation flow. Two applied problems were briefly introduced to demonstrate that very complicated problems, for which the authors have previously struggled to find satisfactory representations using standard approaches and commercial software tools, could be elegantly described as HCCM models.

ACKNOWLEDGEMENTS

The first author thanks the **Austrian Science Fund (FWF): Project Nr. J3376-G11,** for funding his research in New Zealand.

REFERENCES

Balci, O. (1988). The Implementation of Four Conceptual Frameworks for Simulation Modeling in High-level Languages. *Proceedings of the 20th Conference on Winter Simulation*, pp. 287--295.

Birta, L. G., & Arbez, G. (2007). *Modelling and Simulation: Exploring Dynamic System Behaviour.* Springer.

Günal, M. M., & Pidd, M. (2011). *ACM Transactions on Modeling and Computer Simulation:* DGHPSIM: Generic Simulation of Hospital Performance. ACM, (S. 1-22).

Hay, A. M., Valentin, E. C., & Bijlsma, R. A. (2006). Modeling Emergency Care in Hospitals: A Paradox - The Patient Should not Drive the Process. *Proceedings of the Winter Simulation Conference, 2006. WSC 06* (S. 439-445). Winter Simulation Conference.

Lim , M. E., Worster, A., Goeree, R., & Tarride, J. E. (2013). Simulating an emergency department: the importance of modeling the interactions between physicians and delegates in a discrete event simulation. *BMC Medical Informatics and Decision Making, 13*(59).

Mes, M., & Bruens, M. (2012). A generalized simulation model of an integrated emergency post. *Proceedings of the 2012 Winter Simulation Conference (WSC)* (S. 1-11). Winter Simulation Conference.

Overstreet, C., & Nance, M. C. (2004). Characterizations and Relationships of World Views. *Proceedings of the 36th Conference on Winter Simulation* (S. 279-287). Washington, D.C: Winter Simulation Conference.

Pidd, M. (May 1995). Object-Orientation, Discrete Simulation and the Three-Phase Approach. *The Journal of the Operational Research Society, 46*(3), S. 362-374.

Pratt, D. B., Farrington, B. A., Basnet, C. B., Bhuskute, H. C., Kamath, M., & Mize, J. H. (1991). A framework for highly reusable simulation modeling: separating physical, information, and control elements. *Proceedings of the 24th Annual Simulation Symposium, 1991*, (S. 254-261).

Raunak, M., Osterweil, L., Wise, A., Clark, L., & Henneman, P. (2009). Simulating patient flow through an Emergency Department using process-driven discrete event simulation. *ICSE Workshop on Software Engineering in Health Care, 2009, SEHC '09*, (S. 73-83).

Robinson, S. (2006). Conceptual Modeling for Simulation: Issues and Research Requirements. *Proceedings of the 38th Conference on Winter Simulation* (S. 792--800). Monterey, California: Winter Simulation Conference.

Robinson, S., Taylor, S., Brailsford, S., & Garnett, J. (2006). Buildng Communicative Models – A job oriented approach to manufacturing simulation. *Proceedings of the 2006 OR Society Simulation Workshop.*

Robinson, S., Brooks, R., Kotiadis, K., & Van Der Zee, D.-J. (2010). *Conceptual Modeling for Discrete-Event Simulation.* Boca Raton, FL, USA: CRC Press, Inc.

Zeigler, B. (1987). Hierarchical, modular discrete-event modelling in an object-oriented environment. *SIMULATION, 49*(5), S. 219-230.

MICHAEL O'SULLIVAN is a Senior Lecturer in the Department of Engineering Science at the University of Auckland, New Zealand. He received his PhD in Management Science and Engineering from Stanford University. He works on Operations Research (OR) modeling, applications and computation. His current focus is the use of OR to build intelligent cloud systems and improve health care systems. His e-mail address is: `michael.osullivan@auckland.ac.nz`

NIKOLAUS FURIAN is a visiting post-doc at the University of Auckland. He obtained his degree in Technical Mathematics at the University of Technology Graz in 2008. During his PhD program he focused on applied operations research topics, especially bin packing. Currently he working on simulation and optimization applications for health care sector. His e-mail address is: `nfur003@aucklanduni.ac.nz`.

CAMERON WALKER is a Senior Lecturer in the Department of Engineering Science at the University of Auckland, New Zealand. He received his PhD in Mathematics from the University of Auckland. He works on computational analytics, focusing on statistical modelling, simulation and optimization. His e-mail address is: `cameron.walker@auckland.ac.nz`

SIEGFRIED VÖSSNER holds a PhD degree in Engineering Sciences from Graz University of Technology. Until 1999 he was a postdoctoral fellow and visiting scholar at the Department for Engineering Economic Systems and Operations Research at Stanford University, USA. After being a project manager for McKinsey&Company he became professor and chairman of the department of Engineering- and Business Informatics in 2003 and is currently Vice-Dean of the School of Mechanical Engineering and Economic Sciences of Graz University of Technology. E-mail: `voessner@tugraz.at`