Development of Agent-Based Models for Healthcare: Applications and Critique

by

Bryan C.P. Demianyk

A Thesis submitted to the Faculty of Graduate Studies of

The University of Manitoba

in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

Department of Electrical and Computer Engineering

University of Manitoba

Winnipeg, Manitoba, Canada

© Bryan C.P. Demianyk, 2015

University of Manitoba

I. Supervisory/Examination Committee

Supervisory Committee

Dr. Robert D. McLeod, Department of Electrical and Computer Engineering

Supervisor

Dr. Marcia R. Friesen, Centre for Engineering Professional Practice and Engineering Education; Department of Electrical and Computer Engineering

Supervisor

Departmental Examiner

Dr. Ken Ferens, Department of Electrical and Computer Engineering

Outside Examiner

Dr. Rasit Eskicioglu, Department of Computer Science, University of Manitoba

II. Abstract

Agent-based modeling (ABM) is a modeling and simulation paradigm well-suited to social systems where agents interact and have some degree of autonomy. In their most basic sense, ABMs consist of agents (generally, individuals) interacting in an environment according to a set of behavioural rules. The foundational premise and the conceptual depth of ABM is that simple rules of individual behaviour will aggregate to illuminate complex and/or emergent group-level phenomena that are not specifically encoded by the modeler and that cannot be predicted or explained by the agent-level rules. In essence, ABM has the potential to reveal a whole that is greater than the sum of its parts. In this thesis, ABMs have been utilized as a modeling framework for three specific healthcare applications, including:

- the development of an ABM of an emergency department within a hospital allowing the modeling of contact-based infectious diseases such as influenza, and simulating various mitigation strategies;
- the development of an ABM to model the effectiveness of a real-time location system (RTLS) using radio frequency identification (RFID) in an emergency department, used for patient tracking as one measure of hospital efficiency; and,
- the development of an ABM to test strategies for disaster preparedness (high volume, high risk patients) using a fictitious case of zombies in an emergency department.

Although each ABM was purposeful and meaningful for its custom application, each ABM also represented an iteration toward the development of a generic ABM framework. Finally, a thorough critique of ABMs and the modifications required to create a more robust framework are provided.

III. Acknowledgements

First and foremost, I would like to sincerely thank Dr. Marcia Friesen and Dr. Robert McLeod. Without both of their incredible patience, knowledge, guidance, and support, this work would have never been possible. I would also like to thank Dr. Marek Laskowski for the hours of discussions, numerous brainstorming sessions, and development guidance over the years. I must also thank Dr. Ken Ferens and Dr. Rasit Eskicioglu for taking time out of their schedules and providing constructive comments and suggestions in reviewing this thesis. Special thanks to Jesse Vivanco, Julian Benavides, and Ryan Neighbour for participating in various projects over the years and providing an entertaining atmosphere that allowed this research to flourish. Finally, I would like to thank my parents Colin and Carole Demianyk for their encouragement, help, and support in the completion of this thesis.

IV. Dedication

I would like to dedicate this thesis to my entire family. Firstly, to my grandparents James and Jean Demianyk and Eugene and Lillian Waskin, none of who are alive to see the completion of this thesis, but I know they would all be very proud. Secondly, to my parents Colin and Carole Demianyk and sister Alexandrea Demianyk, whose love, support, and encouragement have helped me succeed at whatever I pursue. Finally, to my girlfriend Bárbara López, whose love, patience, and enthusiastic energy have helped pull me through any rough patches encountered while completing this thesis. I love you all.

V. Table of Contents

VI. List of Tables

VII. List of Figures

VIII. List of Acronyms

ABM – agent-based modeling

DES – discrete event simulation

ED – emergency department

GPU – graphical processing unit

RTLS – real-time location system

RFID – radio frequency identification

HPC – high performance computing

ICU – intensive care unit

HCW – healthcare worker

IV – infectious virus

ILI – influenza-like-illness

SIR – susceptible, infectious, recovered

EDIS – Emergency Department Information System

HP – high priority

MP – medium priority

LP – low priority

OLS – ordinary least squares

GP – genetic programming

ZLI – zombie-like-illness

RAII – Resource Acquisition is Initialization

API – application programming interface

1 Introduction and Motivation

1.1 Introduction

Agent-based modeling (ABM) is a relatively new modeling and simulation paradigm. It is well suited to modeling social systems where agents interact and have some degree of autonomy. In their most basic sense, ABMs consist of agents (generally, individuals but can also include non-human and/or non-living entities) interacting in an environment according to a set of behavioural rules. In an ABM, agents interact in manners that are probabilistic or stochastic. In a sense, the agents are interacting as stochastic state machines where the overall behaviour of the system under study is a result of these interactions and behaviours.

The work encapsulated in this thesis originated out of the capstone project in the Department of Electrical and Computer Engineering, University of Manitoba, entitled "Agent-based model synthesized contact graphs for use in disease control" [1]. One of the primary means of spread for contact-based diseases, such as influenza, is the contact among the individuals (agents) in terms of proximity and duration of contact, as well as the disease states of the interacting agents (e.g., infected, recovered, susceptible). From this, numerous modeling refinements evolved to improve the modeling and fidelity of the original ABM used in the capstone project. In this research, several stand-alone research studies were undertaken using an ABM framework for modeling and simulation, and the

objectives shifted to building a more unified ABM framework that would be customizable for a variety of ABM implementation scenarios, rather than building a new ABM framework for each specific modeling application. This thesis includes these studies and more importantly, it critiques the framework and presents alternatives from a software engineering perspective towards a more robust software architecture and improved methods and underlying structures to make ABMs even more useful.

Since the thesis encompasses three stand-alone projects and an extensive critique of an overall ABM framework, it uses previously published work and is presented 'sandwich-style', in that the chapters can be read in a fairly stand-alone way. An ABM is not the only means of modeling stochastic systems. Methods for modeling similar systems include discrete event simulation (DES) and system dynamic simulation. In DES, events can occur asynchronously and the clock skips the next event start time as the simulation proceeds. The system state is a set of variables that capture the system state at any given time. In system dynamics, properties of the whole are modeled as opposed to properties of the elements. System dynamics is more closely related to compartmental differential equation models as a particular instance of a disease spread model. This will be discussed more fully in later sections of the thesis.

In general, ABM is 'bottom-up' systems modeling from the perspective of constituent parts. Systems studied are modeled as a collection of agents (e.g., people in social systems) imbued with properties: characteristics, behaviours (actions), and interactions that attempt to capture actual properties of individuals with a high degree of diversity and

fidelity. In the most general context, agents are both adaptive and autonomous entities that are able to assess their situation, make decisions, compete or cooperate with one another on the basis of a set of rules, and adapt future behaviours on the basis of past interactions. Agent properties are determined by the modeler and are ideally derived from actual data that reasonably describe agents' behaviours (e.g., their movements and their interactions with other agents). The emergence of a data culture, also called 'big data' and associated 'big data analytics', offers new opportunities to use real world data, even in near real time, as inputs into ABMs. The modeler's task is to determine which data sources best govern agent profiles in a given ABM institutional simulation.

The foundational premise and the conceptual depth of ABM is that simple rules of individual behaviour will aggregate to illuminate complex and/or emergent group-level phenomena that are not specifically encoded by the modeler and that cannot be predicted or explained by the agent-level rules. In essence, ABM has the potential to reveal a whole that is greater than the sum of its parts [2],[3].

Agent-based modeling provide a natural description of a system that can be calibrated and validated by representative expert agents, and it is flexible enough to be tuned to high degrees of sensitivity in agent behaviours and interactions. As such, ABMs play a vital role as an information translation vehicle. The lexicon used to develop an ABM is the lexicon of area experts and of the institution under consideration (e.g., a hospital), reflecting the world in a real and as specific a manner as possible. One builds a laboratory where the behaviours of individuals are similar to those in the real-world emergency

department (ED), and one observes what happens when the rules of behaviour and interactions are changed. The underlying ABM engine may be quite complex and utilize the most advanced processing and hardware techniques available, but this level of detail is not required in developing the model or in the analysis of its output.

In this thesis, ABMs have been utilized as a modeling framework for various healthcare applications, including infection spread, patient tracking in hospitals, and zombie modeling as an instance of disaster preparedness. By comparison, the primary mathematical models of disease spread have a long history and are based on coupled differential equations associated with compartments within a population. In the simplest case of a compartmental disease spread model, individuals within a population would be in one of several states (generally less than five), whereas in ABMs, the fidelity is as fine-grained as a state for each agent within the simulation.

1.2 Motivation

This work was initially motivated by a desire to better model infection outbreaks such as the 2009 H1N1 influenza pandemic, as well as more recent outbreaks such as the 2014 Ebola outbreak in west Africa. More specifically, the work here was initially motivated by a more specific scenario of infection spread that may break out within a given institution such as a hospital or workplace. As the research developed, other applications including patient tracking and disaster preparedness were also considered.

It was also desirable to undertake a relatively complex software project that would provide further insights into the technical difficulties associated to ABM as a framework for healthcare modeling. Additional challenges identified early were associated with being able to incorporate as much real data into the simulations as possible, and as a sidebar, a number of applications were investigated that would facilitate a more realistic capturing of agent interactions.

1.3 Objectives

After developing several ABMs for related but different healthcare applications which are purposeful and meaningful in their own right, the objective of the research became to create a generic ABM framework that would overcome some of the shortcomings associated with building one-off ABMs. As such, much of the thesis has been written in a manner to present instances of ABMs that were developed and a critique as to what would need modifying in terms of creating an ABM framework.

Developing an ABM within an object oriented framework from the ground up gives the developer an additional degree of understanding of the ABM technique, in contrast to commercial or alternative open source platforms. By performing the implementation of the ABM oneself, a better insight into the problem is gained. Furthermore, the custom ABM in this work allows for the incorporation of additional data sources on agent behaviour in the future (e.g., wireless sensor networks), which available ABM toolkits do not accommodate, and is created to be more flexible than the domain-specificity inherent

in other available platforms. At the same time, a custom ABM requires an additional time investment in its development compared to the use of a readily available platform, and requires proactive attention to ensuring the inputs of domain experts. When developing code for ABM, there is also an additional advantage in being able to explore computational architectures, such as those based on grid computing or graphical processing units (GPU). Most open source or commercially available frameworks are not as amenable yet to this type of investigation.

Ideally, the ABM framework would simplify the work of the modeler to some degree without sacrificing the advantages of purpose-built ABMs. Emphasis was placed on a framework that:

- does the job it is intended to do;
- is intuitive/easy to use, powerful/fast/efficient;
- is scalable, flexible, easy to extend;
- is easy to maintain, platform independent;

There are a number of ABM frameworks and environments that offer considerable advantages but they also provide constraints that are not easily overcome. In some cases, multithreading and/or cluster-based computing is not possible or is beyond the capability of the framework. Although the final objective is not fully reflected in this thesis, considerable insight into building a suitable ABM framework was acquired.

1.4 Outcomes

The thesis makes several distinct contributions. Firstly, several ABMs and variants where built, furthering recognition of ABM as being a useful modeling paradigm. These include:

- the creation of a sophisticated ABM (erSim) for an ED within a hospital allowing the modeling of contact-based infectious diseases, as well as providing a means of considering several mitigation strategies;
- the redeployment of the hospital ABM to model the effectiveness of a real-time location system (RTLS) using radio frequency identification (RFID) as a means of providing localization information, in this instance, patient tracking;
- the extension of erSim to include considerably more sophisticated agent behaviour in a simulation of disaster preparedness, in this case zombies in an ED and mitigation strategies suitable for that environment; and,
- in recognition of the future requirement of incorporating real data, the author also contributed to demonstrating the role smartphones can play as proxies for people when attempting to build contact graphs associated with contact-based diseases and their spread. Small sections throughout this thesis and cited references illustrate the author's contributions and awareness of merging real individual-based data with ABMs.

Secondly, a critique of ABMs and the modifications required to create a powerful ABM framework are addressed in detail in Chapter 6.

2 Literature Review

2.1 Introduction

This chapter is organized as a general literature review on the evolution and application of ABM technology, and the simulation of complex social dynamics in hospital environments. Hospitals are a promising area of continued ABM research with the potential for substantive outcomes. Healthcare around the world deals with ongoing pressure to find cost efficiencies, and target areas include the optimization of healthcare processes and flow, reducing ED wait times and length of stay, and reduce admission times. Within these areas, hospitals rely on the experience of practitioners for improvements in triage procedures, diverting low-acuity patients, reconfiguring the healthcare staffing model, and reorganizing operational units both physically and procedurally.

Simulation offers a potential to identify improvements and new understandings in how a facility operates. Simulation has the potential to model real-world variability, lessen testing and implementation costs of planned changes, and help minimize the risk of errors in implementing changes. Patient tracking through their stay in the hospital by using technologies (e.g., RFID) and improved electronic reporting and dashboards are examples of initiatives that can be integrated with simulation studies to generate valuable information on social dynamics within the institution.

Although simulation and modeling in healthcare facilities is not new, ABM within these settings is a relative newcomer. This literature survey focusses on hospital ABMs, which is an agent-centric approach as opposed to more established areas of simulation, which tend to the process-oriented. The key differences between modeling techniques such as discrete event, system dynamics, network analysis, and ABM are well documented. To date, the majority of research in healthcare simulation has utilized Monte Carlo, DES, and system dynamics rather than ABMs [4],[5],[6].

Yet, ABMs are considered to be a very promising and complementary technique by which to simulate hospital dynamics. Arguments for their more widespread use within healthcare depend on more widely adopted and effective conceptualization and implementation tools [7]. Some researchers claim that the 'signature' success of ABMs in public health is in the study of epidemics and infectious disease dynamics [6],[8], where the successes of ABMs have demonstrated the importance of the role of social networks, human movement patterns, transportation systems, and the disease dynamic itself. This overwhelming amount of research in applying ABMs to the study of large scale infectious disease spread (e.g., influenza, sexually transmitted infections) is not addressed here. The ABMs applied to institution-scale environments (rather than regional scales) are nonetheless emerging as an excellent vehicle for modeling hospitals due to their inherent ability to leverage social network analysis in a similar manner to social interactions of a large scale infectious disease.

2.2 ABMs within Hospitals

Agent-based modeling has seen a tremendous growth in many areas over the past 15 years, with more recent inclusion of hospital and healthcare settings. The primary application of ABM of hospital environments examines patient flow (e.g., EDs) [9] and other hospital operational issues. This technique is used to examine the dynamics of infection spread within a hospital, such as the hospital's role in an influenza epidemic [10] and the dynamics of hospital-acquired, or nosocomial, infection spread [11]. Economic models of healthcare removed from the scale of patients have also been examined, but these models are not surveyed here.

2.2.1 System Attributes

When designing an ABM for hospital applications, there are choices in system attributes that become design decisions unique to the context and objectives of the model. An ABM is inherently agent-centric, and the model arises from the consideration and definition of the agent's environment, the agent's characteristics, and the agent's interactions with other agents.

- Commercial and homegrown: At present a large number of ABMs are developed as one-offs or custom models, dedicated to the objective at hand. These offer advantages associated with data fusion, accelerators through multicore, cluster, high performance computing (HPC) optimization as well as computation on general-purpose GPU. A disadvantage is the considerable overhead in developing

one's own code, inclusive of code verification. The benefits of a commercial platform are a proven code base and user community. Just as with many other areas where simulation plays a crucial role in product development, eventually the benefits of a commercial product usually outweighs the advantages of a homegrown solution. There are, however, intermediary code bases that are typically open and community supported. These are usually verified to some degree, but usually not to the degree of a commercial offering. All forms of ABM development have associated learning curves. The largest and most popular commercial ABM offering is that within AnyLogic. Open source ABM frameworks include Repast, NetLogo, and Swarm.

- Environment:
    - The topography or layout upon which agents operate is an initial decision in ABM development. Environments can be real-world, synthesized, or abstracted. Real-world environments can be captured from hospital floor plans. The modeler, using simplifications or assumptions as opposed to real floor plans, can generate synthesized environments. The environment can also be abstracted entirely as a data point in the overall model and assigning the agent to discrete non-physical locations within the computer code. However, the strong benefit of ABM is to allow for real-world environments to enhance the validity and credibility of the model, to ease the interpretation of simulation results, and to assist in knowledge transfer.

- Most ABM simulation suites include some means of visualization of the agent within the environment, and this benefit of ABM over other modeling techniques has been accentuated with the affordability and accessibility of high performance desktop computing and graphical processing. Visualization of specific instances of the process allows verification of the model setup, simulation in progress, and simulation results. Where a simulation requires a very large number of iterations to generate meaningful findings, the visualization methods are halted while data are accumulated.

- Agents:
  - The selection of agents is a foundational task of the ABM developer. In most hospital ABMs, the logical selection of agents includes patients and hospital staff members. Basic ABMs for hospital EDs may only include patients, nurses, and physicians [7], while more detailed ABMs include allied healthcare providers who also consult within a hospital, and potentially reaching as far as including visitors and facility personnel not directly involved in healthcare delivery (e.g., maintenance staff). Furthermore, an explicit decision should be made to include or exclude inanimate objects as agents within a hospital ABM. Where the ABM is developed to model infection spread (vs. patient flow), researchers have considered the role of equipment and hospital fixtures as vectors for

infection [12], including medical instruments, bed capacity, allied areas relevant to the main ABM focus (e.g., diagnostic services within an ED ABM). Inanimate agents are modeled without explicit agency or any decision-making capability. Besides their role as vectors in infection spread, the availability and utilization strategies of inanimate agents (e.g., bed capacity, equipment availability) can also be illuminated via ABM.

- The assignment of characteristics or profiles to the agents is another foundational task of the developer. The relevant factors for agent profiles are determined by the objective of the ABM and may include distributions of sex, age and other demographic factors, physical origin and destination within and beyond the topography, and risk factors associated with, infection spread. The power of ABM is accentuated within today's emerging big data culture, where the sources of real data for agent characteristics are numerous and varied. Data sets may or may not have been generated for the purpose at hand. Data sources may include hospital information systems, census data, government databases in the case of publicly-funded health systems (e.g., Canadian Institute for Health Information), cellular service records that can be used to approximate physical trajectories of smartphone users upon a topography, and even smartphone apps that are GPS-enabled. The developer must be aware of limitations and gaps within the data and how those limitations impact the

veracity of the dataset for the ABM's objective. Pre-processing is generally required for a single dataset as well as the consolidation of varied datasets. While data is often technically available, political barriers may restrict access to data. The area of real data is likely the area where ABMs within healthcare facilities will more fully evolve as they install in-house systems to capture the data (e.g., patient flows) themselves, which will support the ability to fine-tune ABMs. Such systems may include electronic records, dashboards, as well as technologies such as RFID. In the case of RFID, both inanimate and animate agents can be tracked.

  - The assignment of rules that govern interactions between agents is the other foundational task of the ABM developer. Here, the ABM's impact is evident in the natural inclusion of expert guidance to establish valid and reliable agent interaction rules, formulated directly in the lexicon of the hospital environment and in the real-world topography of the practitioners, such as nurses and physicians. The role of real data in the assignment of agent behavioural rules is just as significant as is the assignment of agent characteristics or profiles.

- Interventions: Whether the hospital ABM was developed to examine patient flow, infection spread dynamics, or another purpose, the key objective in developing an ABM is to introduce policy changes or interventions (agent profile changes, agent behavioral changes, topography changes, or others) in order to investigate 'what

if' scenarios. In patient flow ABMs, interventions may include topography reconfigurations of the ED or procedural reorganization such as low-priority patient diversions within and between hospitals. In an infection spread ABM, interventions may include agent hygiene behaviours and rules of contact.

- Validation and verification: There are emerging guidelines addressing the importance and techniques to validate ABMs [13] including micro-face validation, macro-face validation, output validation, backcasting to known data, and comparison of output to other modeling methods.

2.3 Early Hospital ABMs

Some early simulation models within healthcare settings were not specifically denoted as ABM but carry all the characteristics of ABM. In 2006, researchers discussed a simulation model of an ED that recognized the strengths of ABM as a means of communication across disciplines. Part of their validation process was consultation with area experts (e.g., doctors and nurses). The model provided a means of evaluating 'what if ' scenarios, specifically, alternative triage methods [12]. Their work was also one of the first to recognize the importance of visualization as a strong element of ABM and the requirement of using as real data as possible, if available.

Earlier in 2001, others argued for the use and requirements of ABM for hospital management, although they do not appear to have built a working model [14]. Another early paper investigated the role of an ABM within an ED, deriving it from a more

traditional intelligent software agent perspective or multi-agent system perspective [15]. That work introduces processes, treatments, individual agents, and protocols for their interaction.

2.3.1 ABMs for Patient Flows

An evolving literature exists with respect to applying ABM, alone or in complement to other techniques, in modeling the operations of EDs. In general, this literature addresses system-level performance dynamics, quantified in terms of patient safety [16], economic indicators [16],[17], staff workload and scheduling [9],[18],[19],[20], and patient flows. While much of the literature addresses system-level operational concerns during periods of typical operation or stasis, there is also literature on modeling healthcare operations during critical incidents, such as disease outbreaks and terrorist attacks [21].

More recently, others have modeled improvements to patient flow using an ABM running on a HPC resource [22]. This ABM was built with NetLogo and representative of the role for which an ABM is well suited. The objective of the study was to provide impact values of alternative policies for patient diversion. As expected, patients that did not require urgent attention and were fast-tracked or diverted improved the capacity of the ED and reduced the length of stay of patients that remained. More extensive consideration of ABMs for patient flow in EDs were developed by the same researchers [23],[24], including the utilization of an ABM within a decision support system for EDs [25]. A contrasting technique to model patient flows would be DES [18]. In similar work,

the role of re-triage in improving ED patient flow is examined [26]. The results are not unexpected and lend additional credibility to the use of ABMs in healthcare modeling by facilitating and modeling 'what if' scenarios. In other work, a pseudo-agent-based approach is introduced into a DES in an attempt to capture the representative strengths of each modeling approach for simulating an emergency department [27]. In that work, the importance of interaction at the agent level is illustrated, not typically captured with DES.

2.3.2 ABMs for Nosocomial Infections

Figure 1 illustrates where modeling nosocomial infections, or hospital-acquired infections would be characterized within the range of healthcare models.

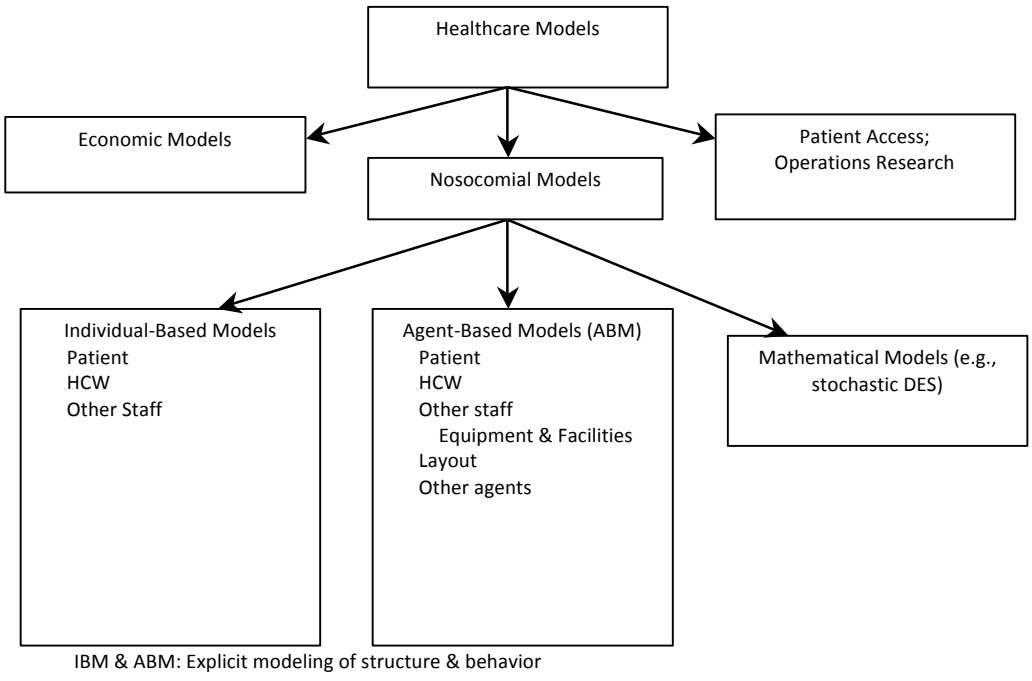


Figure 1. Agent-based nosocomial model within healthcare models

In general, the modeling of nosocomial infections is perhaps the best suited area for ABMs within healthcare institutions. This is largely a consequence of being able to address all of the model components included in Fig. 1 relative to topography and agents. Nosocomial ABMs may be useful in assessing the effectiveness of different infection control protocols or policies, intervention costs, as well as shedding light on potential confinement failures which would accompany widespread infection dynamics [11],[28].

Several of the models oriented to nosocomial infections are known as individual-based models, in which agents are limited by definition to individuals (persons). One such example is a mathematical individual-based model for studying infection spread in a nursing home [29]. By contrast, the notion of an ABM expands the definition of an agent beyond an individual person, to include inanimate objects that can act as vectors of transmission for nosocomial infections. This concept is supported by a significant body of evidence that non-person agents play a significant role as infection transmission vectors [30],[31]. This included the Centers for Disease Control and Prevention's overview of SARS-related information [32] which states, "the virus also can spread when a person touches a surface or object contaminated with infectious droplets and then touches his or her mouth, nose, or eye(s). In addition, it is possible that the SARS virus might spread more broadly through the air (airborne spread) or by other ways that are not now known."

Nosocomial ABM initiatives focus upon explicit modeling of structure and behaviour extending the ABM to include individuals, inanimate objects, and locations. This allows investigation of an organization's policies and practices in the event of a serious nosocomial infection outbreak. Much of the current efforts in nosocomial ABMs set the framework for potential future efforts in modeling and evaluation of organization's documented infection control plans (policies and practices). For example, best practices are available for healthcare practitioners and policy makers dealing with healthcare-associated infections in patient and resident populations. This may be a useful reference to model, as a means of identifying and evaluating their effectiveness. At this time, best practice documents typically reflect "consensus positions on what the committee deems prudent practice and are made available as a resource to the public health and healthcare provider" [33]. Clearly, this is also an opportunity for ABMs to contribute to a collaborative, multi-stakeholder effort.

Despite nosocomial modeling's natural fit with the ABM approach, it is a fairly recent area of exploration for healthcare ABMs [34],[35],[36]. One of the earlier simulation efforts modeled antibiotic resistance in hospitals, contrasting an individual-based model with that of a differential equation model; this included consideration of where the models could be used in conjunction with one another [37]. Another study [11] investigated the spread of a nosocomial pathogen in a dialysis unit using a Monte Carlo individual-based model. The dialysis unit is a very good example of where ABMs may be particularly useful as, "the frequency of patient visits and intimate, prolonged physical

contact with the inanimate environment during dialysis treatments make these facilities potentially efficient venues for nosocomial pathogen transmission" [11]. In a related paper [28], the same authors developed a fairly abstracted nosocomial ABM within an intensive care unit (ICU), advocating for a "conceptually simple discrete element (agent-based or cellular automata) model [that] can explicitly address 'geographic' considerations and probabilistic transmission dynamics germane to the spatially intricate environments and small population sizes characteristic of ICUs". In another nosocomial ABM of an ICU, operational and epidemiological features are considered in an attempt to estimate the effect of understaffing and overcrowding on infection spread [38]. The ABM simulated contact-mediated pathogen transmission that should allow one to establish quantitative relations between patient flow, staffing conditions, and pathogen colonization in patients. Another individual-based approach investigated the role of cohorting, with the aim of minimizing the possible interactions between individuals within a ward [39]. In a relatively recent nosocomial ABM, a combination of differential equation models and probabilistic models are used for each agent in order to simulate changes over time, in bacterial sub-populations within the agent's body [40]. As with many ABM efforts, work is ongoing in terms of validation and verification. In order to construct biologically plausible, transmission-risk models that can guide cross-infection control, researchers have developed an RFID-tracking system in an ED to extract agent-contact data on the understanding of the critical role contact patterns play in cross-

infection control [41]. These types of high-fidelity individual data, topography, as well as contact patterns are ideally suited for an ABM.

2.3.3 Miscellaneous

Other scenarios that have been investigated using ABMs within healthcare settings include optimization of computer terminals [42], serving as another example of incorporating inanimate objects as agents within the ABM. The use of electronic devices including stationary workstations, mobile workstations, tablets, and smartphones in healthcare delivery is evolving very rapidly, and will likely develop momentum that will outpace the insights of ABMs in this area. Other ABMs are oriented to interactions between hospitals where patient diversion on response to load was modeled [43].

2.4 Visualizations

Agent-based models are well suited to visualization as a means of informally verifying and validating the model. Visualization simplifies potentially complex dynamics, providing a better understanding to the recipient of the information. Video-sharing services (e.g., YouTube) are excellent platforms for presenting a model's or project's progress. Figure 2 illustrates a prototype ED. A large waiting area is depicted, where one person is identified at a higher acuity level than the others. Detailed modeling allows for simulation of social distancing within a waiting area, as well as obtaining estimates of ED length of stay as influenced by policy or staffing interventions modeled via ABM.

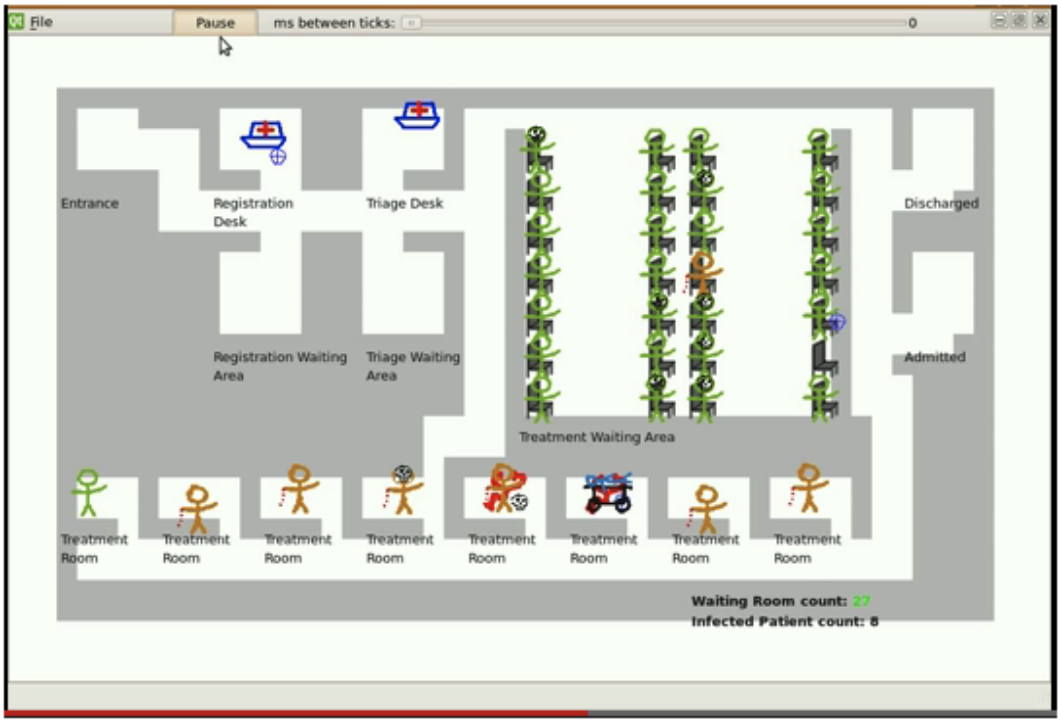


Figure 2. Agent-based model of an emergency department

AnyLogic, a commercial simulation suite of tools has also been fairly widely deployed for ED simulation. An example video has been posted online can be found at http://www.youtube.com/watch?v=LaHdn3GBIWM for public viewing to demonstrate the concept of visualization. A screenshot of this video is shown in Fig. 3.

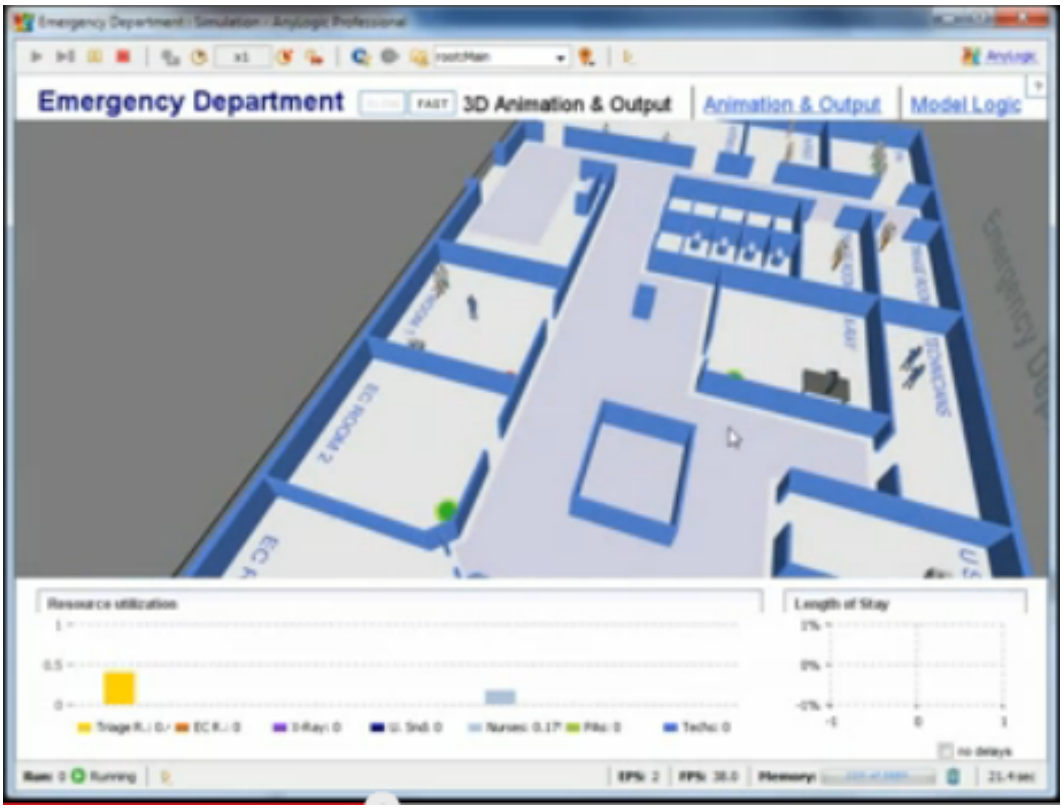


Figure 3. Agent-based model of an emergency department using AnyLogic

It is very likely that hospital simulations and their visualizations will continue to improve, driven largely by initiatives in ‘serious games’ and the gamification of models and simulations. An example of the level of detail one can envision can be found at http://www.youtube.com/watch?v=7CwoMsVyo2Y, which utilized Flexsim, a discrete event simulator. A screenshot of this video is shown in Fig. 4. Similar, high quality animations can be extracted from ABMs.

Figure 4. ‘Gamified’ simulation visualization using Flexsim

2.5 Enhancements to Hospital ABMs

Agent-based models tend to be labour-intensive to develop for each particular application, and are often deployed for specific desired simulations or studies. Although time consuming, they generate vast quantities of data for each simulation run. Typically, the many runs are used to extract statistics that can be used to demonstrate the impact of the infection management policy or intervention being simulated. This massive data generator also offers another benefit: ABMs as potential sources of ‘big data’, to be mined and used in machine learning or pattern classification algorithms. For example, instead of having emergency physicians travel through the ED to see patients in

individual treatment rooms, the patients would travel through the ED to visit (stationary) physicians. Such a policy was generated and validated via a genetic program incorporated within an ED ABM [44].

Another enhancement to ABMs and to simulation in general arises when data analysis augments the simulation. For example, researchers have analyzed data to identify best scenarios extracted from DES of an ED [45]. Although scenarios were extracted from a DES as compared to an ABM, the same type of enhanced data analysis are beginning to emerge in ABM, borrowing heavily from non-parametric methods in operations research. Although ABMs are a useful paradigm for aiding the understanding of a complex system on their own, this significant existing contribution will be augmented with the integration of data analytics.

Agent-based models may also be useful in hospital facility design where additional importance may concern the role of the HVAC system within various departments. This would imply a hybrid of simulation techniques, likely encompassing an ABM and computational fluid dynamics model. In another instance, a hybrid ABM/DES model for emergency medical services in a city is conjectured, although an actual model or simulation has not been reported [46]. Resource planning for placement of RFID readers in an RFID system may be integrated into the ABM to estimate errors associated with the tracking system [47].

2.6 Summary

This chapter reviews the current status of, and advocates for, the increased use of ABMs within healthcare settings, particularly within hospitals. In this context, ABMs of infection spread are among the most advanced and numerous at this time, with an emerging body of work associated with ABMs investigating patient flow and other operational processes in hospitals.

Agent-based models are not without their disadvantages. The difficulty in developing robust validation and verification techniques is acknowledged by the ABM research community; this is a difficulty faced by many simulation modalities. Other difficulties arise from the challenge of generating accurate models of agent behaviours and interactions, as well as data extracted from the systems being modeled. The role of ABMs as useful simulation vehicles within healthcare facilities is still in its infancy, but offers tremendous potential for the better understanding and optimization of these complex systems. The emergence of ABMs will likely evolve towards a more integrated simulation and analysis suite, combining with other established techniques.

3 ABM 1: erSim

3.1 Introduction

The following chapter describes work done as part of a research contract with the Public Health Agency of Canada. Outlined is an ABM to model the spread dynamics and potential mitigation measures for hospital-acquired (nosocomial) infections in hospitals. The ABM software development was in collaboration with Dr. Marek Laskowski, at the time a Ph.D. student in the same laboratory. Information in this chapter will serve as a basis for the development of further ABMs described within the thesis. As mentioned in the literature review, the use of ABMs within a hospital or ED has been gaining traction in terms of utility and will continue to do so as data-driven models continue to evolve [48]. A version of this chapter was previously published [10]. The author's contributions and responsibilities were to:

- fix any inherent problems with the current simulation that caused it to crash;
- improve simulation controls (e.g., speed, zoom, pause);
- add new agent types (e.g., chairs);
- introduce interactive model building;
- import/export layout or configuration files;
- improve collection and extraction of more specific data;

3.2 Framework Implementation

The ABM framework was extended from a preliminary framework reported in [49]. It was written in the object-oriented language, C++, using Qt4 libraries running under the Linux operating system. This approach has natural extensions to the sort of spatial modeling called for by the spatial nature of the system under study. The simulated world is a two-dimensional discrete Cartesian plane of extremely high resolution (floating point). The model is not general purpose, but is a spatially explicit ABM aimed at institution-level simulation of a healthcare environment. In this case, the model reflects a prototypical ED layout based on a representative ED in Winnipeg, Canada.

Within an object-oriented approach, all agents and other entities in the system are implemented as objects with a geographical hierarchical decomposition. For partitioning the simulated world further, there is a discrete grid overlaying to the world's Cartesian plane; however, the underlying space is still 'continuous' (floating point specification of x and y coordinates). The simulation loop calls a 'tick' method, the semantics of which are that the object (agent) is getting its slice of simulation time where it can react to other agents around them. Agents can access a limited number of features of other agents and objects, which can be set by the programmer, depending on the model. They can also pass messages to other agents in order to achieve interaction. For example, they can 'sleep' until bumped or acted upon. Another difference between this and other modeling approaches is the lack of a scheduler, which drives the simulation. Agents spend most of their time reacting to one another, so that the scheduler would effectively be activating

each agent at every time step. For example, on any particular ‘tick’ or time step, a patient can assess the number of patients waiting ahead of them, and decide whether or not to leave the ED without treatment, based on their condition and a function of how long they will have to wait. It is acknowledged that a scheduler or script may be beneficial to the efficiency of this ABM, especially when extending the model to simulate more complex time-of-day variation of patient arrivals and staffing schedules. Partitioning the space of the world can also be used to reduce the number of agents that each agent has to interact with at each time step, thus reducing overall simulation time.

In this framework, agents include patients, healthcare workers (HCWs) (e.g., doctors and nurses), and inanimate objects (e.g., chairs) that can act as transmission vectors. Each agent acts in a circular order using a set, predictable, but arbitrary order in which they were added to the simulation. In order in which agents were added is governed by the stochastic arrival rates of patients of various triage scores. Actions taken by the agent within its ‘tick’ method may change the internal state of the agent (e.g., position within the world). Within the ‘tick’ method, agents react to their current internal state as well as the state of other agents currently within spatial proximity. Time is discrete with each time step having a resolution of one second. Therefore, a particular agent A_i will be reacting to some agent states that have come before A_i in the order (and have already reacted to the current state of A_i) and some agent states that come after A_i (and will react to the new state of A_i when their turn comes in order). Alternatively, agent states could be held fixed until each agent has a chance to react to every other agent’s current state. The

latter mode of operation is analogous to ordinary cellular automata, while the former mode of operation is analogous to sequential cellular automata [50]. Both modes are valid methods of simulating complex systems. In any case, the resolution of the time slice is appropriate with respect to the duration of most agent contacts, as well as the duration of the simulation. An experiment was done to verify this assertion: the order in which the agents have their 'tick' methods called was shuffled between every time step, and there was no statistically significant difference in the results. The most computationally straightforward mode was chosen for implementation, being the aforementioned sequential mode.

Patients are modeled as occupying a circular space with a radius of 60 cm (30 pixels on screen), representative of their physical person as well as a concept of personal space. Close contact is defined as any interaction where the centre-points of the agents are within 40 cm (20 pixels on screen) of one another. Casual contact is defined as any interaction where the centre-points of the agents are 40–120 cm (20–60 pixels on screen) apart. These choices were arbitrary but not unreasonable, and are easily modified as parameters in the scenarios discussed in this chapter. The distance between agents is a function of the nature of interaction between agents. Contact between HCWs and patients will be close contact for the triage and treatment processes, and casual contact for the registration process. A doctor treating a patient is considered close contact. Contact between patients will typically be casual contact, with the exception of particularly crowded waiting rooms. Contact between HCWs is modeled as casual contact.

Differentiation between contact types becomes relevant in the discussion of various spread probabilities of infectious viruses (IVs) in the following section. Figure 5 illustrates several configurations of agent interaction.

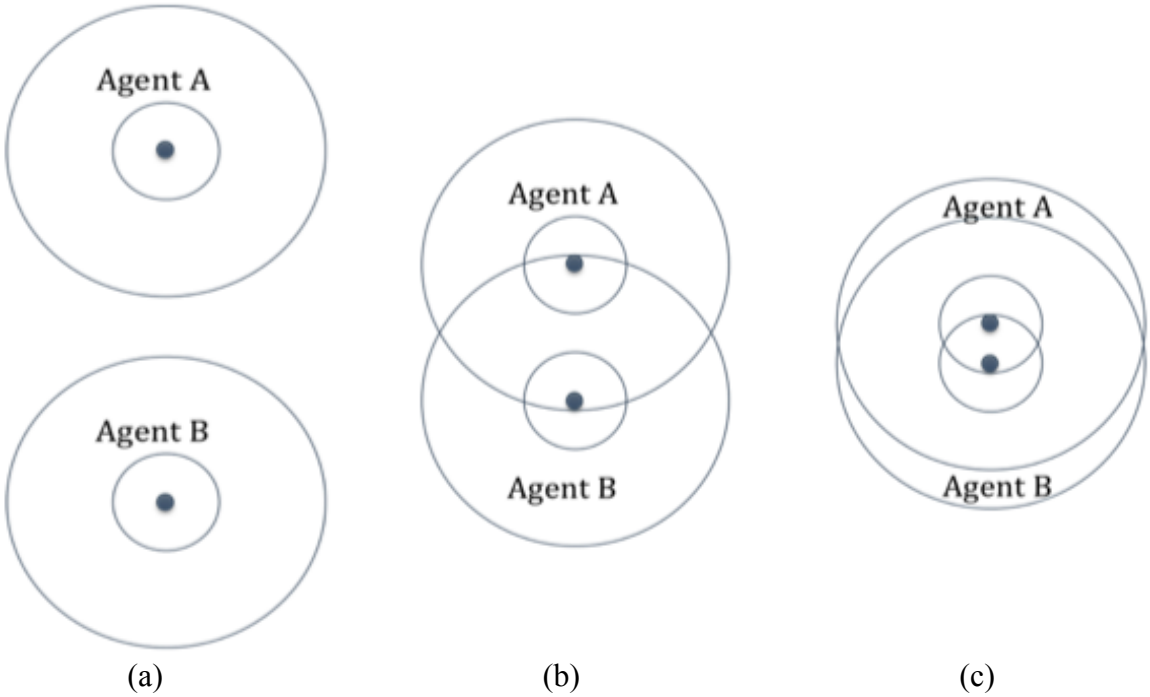


Figure 5. Agents in: a) no contact, b) casual contact, c) close contact

Using this framework, a hospital ED was modeled as a collection of interacting agents. Patients arrive and require treatment. Nurses and other agents decide upon treatment plans for patients. Doctor agents treat patients. Agents practice rudimentary path planning, typically choosing the shortest path to the destination location (e.g., travel from waiting room to treatment room). The agents also selectively practice microsocial-

distancing, where they will tend to distance themselves according to their local density, for example, when selecting a chair in the waiting room. These are examples of localized agent decision-making. Figure 6 illustrates a conceptual patient flow. Further details about the basic patient flow model used are provided by [49].

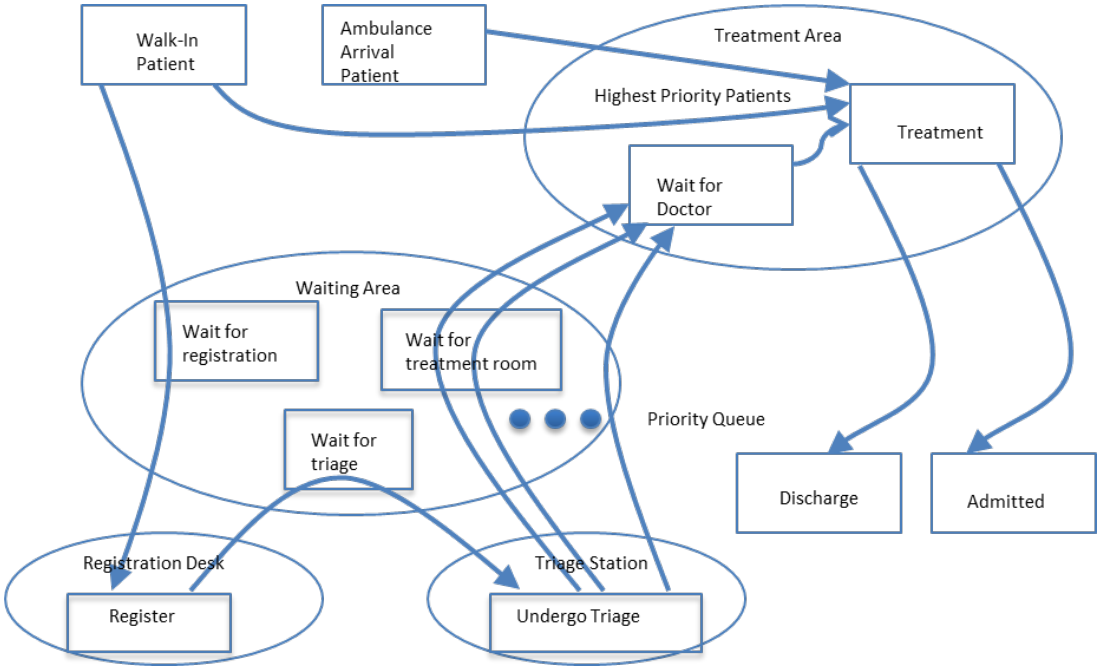


Figure 6. Conceptual patient flow through the ED

Throughout the patient flow described in Fig. 6, agents come into contact with one another. During these interactions, infection spread is modeled, primarily driven by agent distance during contact and the duration of the contact. The probability of transmission is based on agent distance per time step found in Table I. The basic disease function for modeling an influenza-like-illness (ILI) is described in Fig. 7. Although clearly inspired

by classical epidemiological models (e.g., SIR), this model differs in several ways and constitutes a distinct contribution.

```
Let CLOSE_RATE := ILI Close Transmission
   Probability Per Second
Let CASUAL_RATE := ILI Casual
   Transmission Probability Per Second
For each Infectious Agent A in World:
        For each Uninfected Agent B within
    Casual Range of A:
                Generate random number R between 0 and 1
                If B is within Close Range of A:
                        If R < CLOSE_RATE: A infects B
                Else:
                        If R < CASUAL_RATE: A infects B
```

Figure 7. Pseudocode for ILI infection spread

Agents that enter the simulation infected with an IV are modeled as infectious upon entering (ILI-presenting). Agents that acquire the IV while visiting the ED are infectious after three hours have passed. An important aspect of this model is agent immunity. An agent can be modeled as immune to the IV, in which case they cannot acquire the infection and accordingly cannot spread the infection. Chairs are modeled as potential IV vectors, however, no in-depth analysis was conducted on their effect. For the current stage of ABM development, this simplified infection model was used and is intended to be representative of a non-specific pathogen. As the ABM is refined in terms of agent behaviours and parameters, additional complexity in the epidemiology can also be incorporated. For example, distinctions between agents' infected, symptomatic, and

infectious states (time periods) can be more clearly drawn; immunity can be varied by age of patient, and virus-specific parameters fine-tuned.

Since ABM is computationally intensive, especially with a temporal resolution of one second, parallelism should be used wherever possible. Parallelism was implemented during the collection of simulation results, as the ABM can be tailor-made to exploit both fine-grained and coarse-grained (process level) parallelism. Coarse-grained parallelism is considered to be the distribution of individual runs to separate processes, whereas fine-grained parallelism is understood as distributing multiple processes within individual runs to multiple processors where possible. Coarse-grained parallelism was exploited here where individual ABMs were run on a computing cluster utilizing 160 CPUs. Since the number of agents concurrently simulated is relatively small, as is the size of the simulated ED, the memory requirements were under 100 MB for each instance of the simulator. Yet, approximately 1200 CPU hours were needed to perform the simulation runs discussed in this chapter, which would have taken about 50 days running serially on a single-processor workstation. This justified the exploitation of using parallelism. Future efforts will focus on moving toward fine-grained parallelism.

3.3 ABM Simulation Parameters

The ED layout corresponded to an area of approximately 21 by 28 metres (1080 by 1400 pixels on screen). While the scaling is not precise, it was chosen to be a reasonable representation. The basic layout was synthesized to be representative of a typical ED in

Winnipeg, Canada and was comprised of a registration and triage area, a waiting room, and multiple treatment rooms accessed by corridors. The floor plan layout used, and corresponding user interface for visualization is shown in Fig. 8.

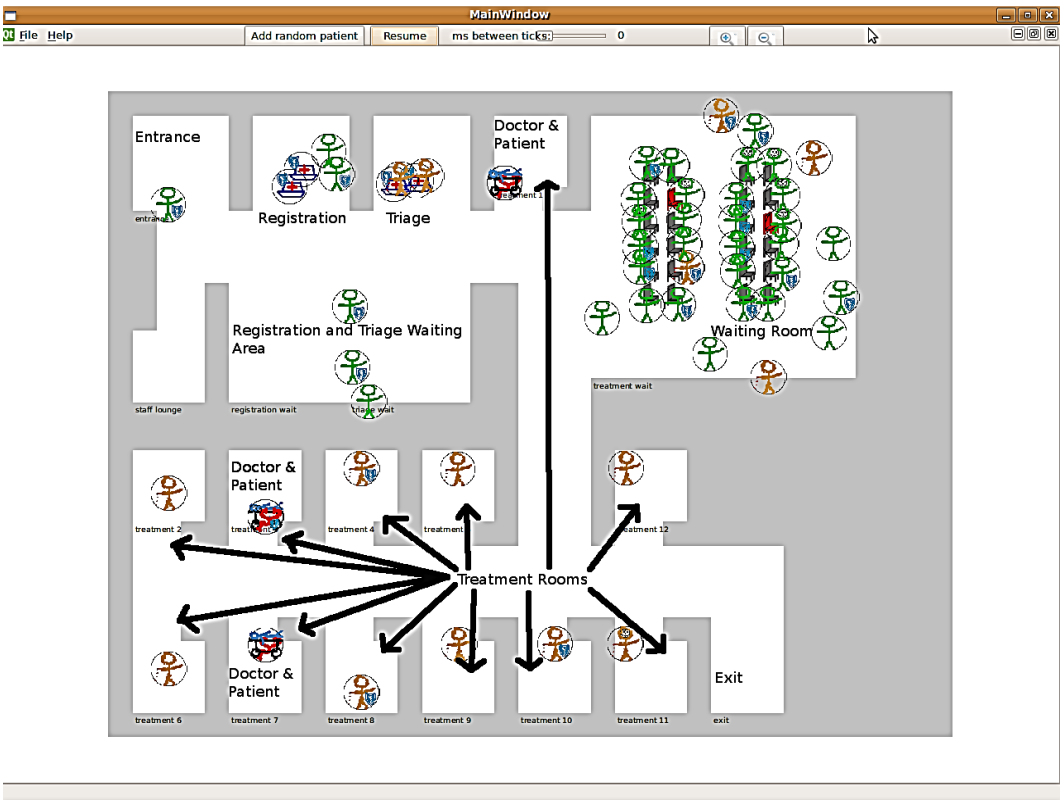


Figure 8. Graphical layout of an emergency department

Table I summarizes the parameters that were varied, with the baseline or nominal case shown in parentheses. A total of 28 cases (scenarios) were individually simulated. The baseline or nominal case is the first of 28 scenarios. The other 27 scenarios consisted of varying one parameter at a time from the baseline or nominal case. Throughout this

section, parameter choices were developed using references from literature [51],[52] and data provided by anonymized Emergency Department Information System (EDIS) data from the Winnipeg Regional Health Authority [53].

TABLE I SIMULATION PARAMETERS AND VALUES

Parameter	Parameter values (baseline value)
Waiting room capacity	25, (50), 75
Number of triage nurses	1, (2), 3
Number of registration nurses	1, (2), 3
Number of doctors	2, (3), 4
Uninfected high priority patient arrival rate (patients/hour)	0.00900462, (0.01800925), 0.03601851
Uninfected medium priority patient arrival rate (patients/hour)	0.7067844, (1.4135652), 2.8271304
Uninfected low priority patient arrival rate (patients/hour)	1.150326, (2.3006484), 4.6012968
Infected high priority patient arrival rate (patients/hour)	0.00210648, (0.00421296), 0.00842592
Infected medium priority patient arrival rate (patients/hour)	0.1378944, (0.2757888), 0.5515776
Infected low priority patient arrival rate (patients/hour)	0.2492136, (0.4984272), 0.9968544
Patient ILI immunity (probability of patient's initial condition)	0.0, (0.50), 0.75
HCW ILI immunity (probability of HCW's initial condition)	0.0, (0.50), 1.0
Close and casual contact ranges in cm	(40 and 120), 80 and 240
ILI transmission probability per second, casual and close	0.00005 and 0.0001, (0.0001 and 0.0002), 0.0002 and 0.0004

Waiting room capacity affects the room size in which the patients wait for treatment before they are assigned a treatment room. The waiting room capacity roughly

corresponds to the most patients that will fit in the waiting room based on relative size. Capacities were chosen to provide a reasonable range of values for a given patient flow.

The number of triage nurses affects how many patients can concurrently be serviced at the triage desk. Increasing the number of nurses has the standard queuing theory implication of reducing the expected wait time of the enqueued patients. However, in this model each nurse is a potential new contagious IV carrier. The number of registration nurses function as above for the registration desk and related services.

A number of factors affect how many patients can undergo the treatment process concurrently. This is conceptually analogous to varying the number of nurses, as above. The baseline was chosen to approximate handling the patient flow, and be representative of typical staffing in an urban ED. The number of doctors was both increased and decreased to assess the effect of this parameter.

To model a surge of patients presenting with ILI, arrival rates of both infected and uninfected patients were obtained using the EDIS. Patient arrival rates for an anonymized hospital with similar characteristics to the ED portrayed in Fig. 8 were mined from data available for the peak H1N1 pandemic month of November in 2009. These rates correspond to approximately five patient arrivals/hour (both uninfected and infected). Arrivals are modeled as six separate Poisson arrival processes; three each for both uninfected and infected patients, varying by priority level (low, medium, and high). Corresponding arrival rates (patients per hour) were divided by 3600 to obtain patient arrivals per second. Patient priority (low, medium, high) affects the order in which

patients are treated within the ED, with high priority (HP) patients being seen sooner than medium priority (MP) and low priority (LP) patients. Patient priority does not impact survivability, because patients do not die within the simulation period of the model. Simulated scenarios included doubling and halving these rates, one perturbation at a time. This type of data is expected to become more readily available here as well as elsewhere, as advances in electronic record keeping proceeds and availability and sharing of this type of information improves [54].

Patient immunity rates are the same across all patient categories, and dictate the probability of any uninfected patient being immune to an IV infection upon arrival. An initial probability of 0.5 indicates that half of the arriving uninfected patients will enter the simulation immune to an IV infection. This probability is reduced but only increased to 0.75 for the simulated scenarios; doubling the probability to 1 would trivialize results, as all patients are immune and none would become infected. Although the probability of patients entering immune can be varied anywhere between 0 and 1, the immunity of individual patients is a binary concept, either they are completely immune, or not at all.

Similar to the patient immunity, HCW immunity dictates the probability that any given HCW is immune to an IV infection at the beginning of the simulation. Unlike patient IV immunity, the HCW IV immunity was also investigated at 1, as this is a probable scenario, particularly in the event of epidemic or pandemic influenza where staff would receive preventative vaccinations. Neither patients nor HCWs can acquire immunity during the course of the simulation.

As discussed earlier and illustrated in Fig. 5, the close and casual contact ranges dictate the spacing between agents, for various agent interactions that take place. Despite agents spacing themselves at close or casual contact range depending on the nature of the interactions between the agents, the arbitrary choice of distances for close and casual contact ranges may have a second-order effect on short-term incidental contact between agents such as in the waiting room. Therefore, the close and casual contact ranges were both doubled from the baseline values to assess the effect of this parameter.

The final parameter, the IV close and casual transmission probabilities are the probability that on any given time step, an infectious agent will infect a particular uninfected agent within close and casual contact range, respectively. The casual transmission probability is one-half that of the close transmission probability as suggested by [29]. The specific values used here were chosen such that some, but not all, agents became infected during a prototypical simulation.

Other potential features include providing the patient with the decision-making capability of leaving untreated. Data on patients that left without treatment are also available and were extracted from EDIS information. In the case of the Winnipeg Regional Health Authority, hospital EDs post the priority of patients and their number as well as expected time to treatment; this has the effect of promoting attrition in the case of low priority patients anticipating long waiting times. Again, this is a stochastic process providing agents with a small degree of decision-making capability and autonomy.

The model underwent initial validation through ‘corner cases’ that verified basic intuitive outcomes, but owing to their arguably unrealistic assumptions are not included in the results presented in the next section. ‘Corner cases’ included, but are not limited to, simulations in which all agents are immune (no acquired infections were observed) and simulations in which all uninfected agents are not immune, and the transmission probabilities are 1 (all uninfected agents acquire the infection).

3.4 Results

The results section was largely a collaborative effort between Marek Laskowski, Julia Witt, Marcia Friesen, and myself, and is summarized here for completeness. The author carried the primary responsibility for ABM development, modifying the erSim ABM to support collection of statistically significant data, and investigating policy efficacy. This is again one of the benefits of using and ABM within a team of persons with disparate skills, whereby the ABM and familiar lexicon facilitates multidisciplinary research.

Based on Table I and the ED floor plan shown in Fig. 8, 28 scenarios (varying one parameter at a time from the baseline or nominal case) were simulated 1000 times each, with roughly 700 patients visiting the ED each run.

A time period of 168 hours (seven days) was simulated, with the first 144 hours (six days) considered a ‘warm-up’ or calibration period. This calibration period is very similar to that used by [35]. During this calibration period, (uninfected) patients arrive and are treated as described previously, identical to the way the ED behaves during the

simulation period (hours 144 through 168), except data are not collected during the calibration period. The purpose and choice of duration of the calibration period was to fill the ED with some reasonable level of patients before data collection begins, as an empty ED is an extremely unlikely occurrence [53]. The calibration period also allowed any potentially occurring transient phenomena to die down, in order for the simulation period to begin with the ED in a baseline condition.

At the end of the calibration period, all HCWs were replaced with uninfected doctors and nurses (as Table II suggests), although the infected patients in the ED remained. For data collection, hour 144 was considered the start of the simulation and hour 168 was considered the end (i.e., the seventh day).

The simulation time period was specifically chosen to model the functionality of the ED during a short period within an ILI surge. The one-week simulation period and the one-day data collection period can be viewed as random samples within a longer period of ILI surge. The simulation was not intended to model ED operations over an entire influenza season. The infection mitigation strategies simulated within the ABM also represent first-line responses to ILI surges that are amenable to short-term and immediate implementation, rather than long-term strategies for community-level infection control. As stated in the objectives of the work, the results are specific to the instance context of the ED being simulated, but may lend general guidance to the ED operations in other instance contexts.

The variables of interest in the simulation were the IV infection probabilities among HCWs and among patients as a result of their contact with one another during the simulation. Specifically, the recorded output data included the following:

- patient count at 144 and 168 hours;
- infected patient count at 144 and 168 hours;
- HCW count at 144 and 168 hours;
- infected HCW count at 144 and 168 hours;
- count of infected patients entering the simulation between 144 and 168 hours;
- count of all patients entering the simulation between 144 and 168 hours;
- count of infected patients leaving the simulation between 144 and 168 hours; and,
- count of all patients leaving the simulation between hours 144 and 168 hours.

These data were used to generate the dependent variable as the percentage of patients in the ED who became infected between 144 and 168 hours in the simulation. The data were stratified into two separate, non-overlapping samples. The first sample included the simulations of 28 individual scenarios in which the baseline parameters were changed one at a time from the nominal or baseline case (‘parameter changes’). The second sample included the simulations of four infection mitigation policies and the baseline values for the ‘four policies’. An ordinary least squares (OLS) regression was used to analyze both the ‘parameter changes’ data and the ‘four policies’ data, in each of which the dependent variable was the percentage of patients in the ED between 144 and 168 hours who became infected during that time.

The first analysis (using only the data with the baseline parameter changes) helped identify variables that were effective in reducing the number of patients that became infected in the ED. From these baseline results, 'four policies' were formulated for further simulation, in terms of their effectiveness in reducing the number of patients that became infected. For example, since patient immunity was found to be important, a policy where all ILI-symptomatic triage patients are masked was simulated. The 'four policies', their formulation and the results of the analysis are detailed following the presentation of results from the 'parameter changes' set of simulations.

Independent variables in the 'parameter changes' OLS regression included parameters initially set at their baseline value, then varied one by one, using the low and high values of the range given in Table I, to measure the effect of those changes. All independent variables, variables used to create the dependent variable, and other variables in the model, as well as their means and ranges, are shown in Table II. There are two sets of numbers: one for the 'parameter changes' analysis and one for the 'four policies' analysis. Since the data sets used in each of these analyses are mutually exclusive, the means and ranges are given for each data set. From Table II, a high degree of variability is evident in the range of variables, highlighting the ABM approach with its inherent focus on transient phenomena.

TABLE II SUMMARY STATISTICS OF VARIABLES IN EACH MODEL

Variable	Parameter changes			Four policies		
	Mean	Min.	Max.	Mean	Min.	Max.
Number of patients that become infected between 144 and 168 hours	14.52	0	102	7.52	0	41
Percentage of patients that become infected between 144 and 168 hours	11.76	0	58.06	6.84	0	32.03
Infected HCW count at 144 hours	0	0	0	0	0	0
Infected HCW count at 168 hours	2.63	0	8	2.57	0	11
Number of patients entering between 144 and 168 hours	111.0	42	244	108.27	67	145
HCW count at 144 hours	7	6	8	6.97	5	8
Patient count at 144 hours	28.54	0	147	22.58	0	62
Patient ILI immunity	0.4875	0	0.75	0.5	0.5	0.5
HCW ILI immunity	0.50	0	1	0.5	0.5	0.5
LP infected arrival rate	0.51084	0.24912	0.99684	0.4986	0.4986	0.4986
MP infected arrival rate	0.2826	0.13788	0.55152	0.27576	0.27576	0.27576
HP infected arrival rate	0.00432	0.002106	0.008424	0.004212	0.004212	0.004212
LP uninfected arrival rate	2.358	1.1502	4.60116	2.30076	2.30076	2.30076
MP uninfected arrival rate	1.449	0.70668	2.82708	1.41372	1.41372	1.41372
HP uninfected arrival rate	0.016884	0.004212	0.018	0.018	0.018	0.018
Casual contact range, pixels	63	60	120	60	60	60
Close contact range, pixels	21	20	40	20	20	20
Number of doctors	3	2	4	3	3	3
Waiting room capacity	50	25	75	50	50	50
Number of registration nurses	2	1	3	2	2	2
Number of triage nurses	2	1	3	2	2	2
Close transmission probability	0.000205	0.0001	0.0004	0.0001	0.0001	0.0001
Casual transmission probability	0.000105	0.00005	0.0002	0.0002	0.0002	0.0002

Note that the maximum 'Infected HCW count at 168 hours' in Table II exceeds the staff count at any one time from Table I. This reflects the policy of sending infected staff home, and replacing them with fresh, uninfected staff. In the worst-case scenario, all HCWs (in the three HCW groups: triage nurses, registration nurses, and doctors) are infected and are waiting for replacements, and the replacements (one replacement for each of three groups) become infected immediately upon arrival.

3.5 Results of Parameter Changes

Results for the OLS regression comparing changes in the baseline parameter values ('parameter changes') are shown in Table III. The R^2 for the model is 0.4543. The results show the effect that a one-unit change in each of the independent variables had on the percentage of patients that became infected between 144 and 168 hours in the ED.

Healthcare workers, particularly doctors, in the ED have a significant effect on the percentage of patients that became infected. For every one-unit increase in the number of doctors and triage nurses, the percentage of patients that become infected decreased by 7.220 and 0.556, respectively. However, every additional registration nurse increased the percentage of patients that became infected by 0.209, although this was only significant at the 10% level.

TABLE III OLS REGRESSION RESULTS FOR 'PARAMETER CHANGES'

Variable	Coefficient	SE	*p*-value
Patient ILI immunity‡	-23.421	0.610	< 0.001
HCW ILI immunity‡	-10.113	0.208	< 0.001
Casual contact range‡	0.127	0.005	< 0.001
HP infected arrival rate‡	2,644.9	30.0	< 0.001
HP uninfected arrival rate‡	-709.8	16.9	< 0.001
Number of doctors‡	-7.220	0.117	< 0.001
Waiting room capacity‡	-0.056	0.005	< 0.001
Number of registration nurses†	0.209	0.123	0.088
Number of triage nurses‡	-0.556	0.127	< 0.001
Close transmission probability‡	38,683	906	< 0.001
Constant‡	38.022	0.774	< 0.001
R^2	0.4543		
Number of observations	20,000		

‡Significant at the 1% level; *significant at the 5% level; † significant at the 10% level

In terms of immunity, patient immunity had a larger effect than HCW immunity: a one-unit increase in patient immunity reduced the percentage of patients that became infected by 23.4, whereas an equal increase in HCW immunity reduced that percentage by 10.1. Technically, this result is the statistical interpretation of the coefficient (i.e., the percentage change in infected patients due to a one-unit change in the independent variable of patient immunity). However, it should be noted that the patient immunity variable was constrained between 0.0 and 0.75 in the simulation (Table I). Thus, simulation results would not actually exhibit a 23.4 percent reduction in infected patients, but would exhibit a percentage reduction in infected patients extrapolated between 0 and 23.4.

An increase in the close transmission probability by 0.0001 increased the percentage of patients that became infected by 3.87. Since the casual transmission probability equals the close transmission probability divided by two, and the two probabilities were varied together, it was not included in the regression. An increase in the casual transmission probability of 0.0001, however, led to a 7.74 increase in the percentage of patients that became infected. A 0.001 unit increase in the HP infected arrival rate increased the percentage of infected patients by 2.6; a similar sized increase in the HP uninfected arrival rate decreased the percentage of infected patients by 0.71. Again, LP and MP arrival rates were not included in the regression since changes in these rates were equivalent to changes in the HP arrival rates and they were effectively multiples of one another. Increases in the LP and MP arrival rates would change the coefficients on those variables (and hence, on the percentage of patients that would become infected) by the inverse of these multiples. A one-unit increase in the casual contact range increased the percentage of infected patients by 0.127. The close contact range equaled the casual contact range divided by three, and so a one-unit increase in the close contact range led to an increase in the percentage of infected patients by 0.381. Finally, if waiting room capacity increased by one unit, the percentage of infected patients decreased by 0.056. In the regression that was not bootstrapped, the number of registration nurses was not significant ($p > 0.10$).

Additional analyses of the variables produced some other interesting results. Controlling for the number of patients in the ED at 144 hours absorbed some of the effect

of doctors. The number of doctors reduced the percentage of patients that became infected because having more doctors meant fewer patients were waiting in the waiting room. But at a given level of patients in the ED at 144 hours, increasing the number of doctors had a much smaller effect on the percentage of patients that became infected in the ED (coefficient = -1.92, $p < 0.001$), because doctors were also transmitting infections at a closer range (doctor-patient) than other patients (patient-patient). Further evidence of this effect can be found in an analysis where the dependent variable is the number of patients that became infected between 144 and 168 hours, and the number of patients in the ED at 144 hours was included in the regression (to control for initial crowding conditions in the ED). In this case, doctors had a significant positive effect (coefficient = 2.83, $p < 0.001$). This suggests that, in absolute terms, for a given number of patients waiting in the ED at 144 hours, when more doctors saw more patients over the following 24 hours, the doctors spread infections. Additionally, controlling for the number of infected HCW at 168 hours switched the sign of the coefficient on the number of registration nurses; that is, an increase in registration nurses reduced the percentage of infected patients, when infected HCW at 168 hours entered the regression separately (coefficient = 0.220, $p = 0.038$). This suggested that, whereas the benefit of having an additional registration nurse for a given number of infected HCW at 168 hours was relatively small, increasing the number of registration nurses, in general (that is, without separately accounting for infected HCW at 168 hours in the analysis), led to increases in the spread of infections.

3.6 Results of Four Policies

The OLS results from the 'parameter changes' analysis identified which independent variables were statistically significant in the percentage of patients that became infected between 144 and 168 hours (Table III). Based on these results, 'four policies' were considered for further simulations. The rationale for the policies is outlined below, and the results of the OLS regression comparing the four policies are shown in Table IV.

TABLE IV OLS REGRESSION RESULTS FOR 'FOUR POLICIES'

Variable	Coefficient	SE	*p*-value
Policy: mask infected HCW‡	-2.288	0.225	< 0.001
Policy: dismiss infected HCW‡	-2.380	0.203	< 0.001
Policy: mask low priority patients‡	-3.159	0.230	< 0.001
Policy: dismiss low priority patients‡	-5.081	0.199	< 0.001
Constant‡	9.421	0.164	< 0.001
R^2	0.1142		
Number of observations	5,000		

‡Significant at the 1% level; *significant at the 5% level; †significant at the 10% level

The four intervention policies model various infection control practices that could be implemented in an ED, with the objective of determining whether any of the interventions led to statistically significant outcomes of the dependent variable. The four polices were simulated one at a time (independently), using the 'baseline value' parameter specified in Table I, on the floor plan featured in Fig. 5. Thus, four distinct

simulations (scenarios) are added to the 28 scenarios investigating ‘parameter changes’, for a total of 32 scenarios. Investigating combinations of policies is left for future work.

Firstly, because patient immunity was statistically significant with a relatively large effect (Table III), the modeled intervention was the triage nurse masking all ILI-symptomatic patients. Within the ABM, this was modeled as reductions to the IV transmission probabilities (close and casual contact) for LP patients once they reached the triage stage. If an patients was masked and infected, then the probability of the masked agent infecting another agent was scaled by 0.5 for casual contact, and scaled by 0.75 for close contact. Therefore, the masked patient's probability of infecting an agent within casual contact range (per time step) was: 0.5*0.0001 = 0.00005 and 0.75*0.0002 = 0.00015 within close contact range. Similarly, the probability of an infected, unmasked agent infecting a masked patient (per time step) is: 0.5*0.0001 = 0.00005 for casual contact, and 0.75*0.0002 = 0.00015 for close contact. An infected masked patient's probability of infecting an uninfected masked patient (per time step) is: 0.5*0.5*casual transmission probability = 0.25*0.0001 = 0.000025 if they are within casual contact range and, 0.75*0.75*close transmission probability =0.5625*0.0002 = 0.0001125 if they are within close contact range.

Secondly, because the close transmission probability was statistically significant (Table III), and because further analysis showed that, *ceteris paribus*, more HCWs led to an increase in the number of infections. The modeled intervention was to mask all HCWs. This was similarly modeled as reductions to the IV transmission probabilities (close and

casual contact) for HCWs. The HCW masks were modeled identically to the patient masks mentioned previously, with the exception that HCW begin the simulation masked, if the policy was in effect.

Thirdly, because the HP infected arrival rate was statistically significant (Table III), the modeled intervention was to direct ILI-symptomatic patients into an alternate treatment stream or area, as could be done during an influenza epidemic. This intervention effectively cohorted these patients by removing them from the rest of the patient population. Unlike the masking policy, this policy was only applied to infected LP patients. The triage nurse identified these patients and then removed them from the simulation (i.e., sent them home or to an isolated treatment area through a regular exit route). The model for this policy could be further elaborated by adding a probabilistic misdiagnosis model whereby infected patients are probabilistically allowed to remain in the normal treatment stream. Similarly, uninfected patients may erroneously be sent home.

Finally, additional analyses indicated that the number of infected HCWs at 168 hours and the close transmission range were statistically significant (Table III). In conjunction, because further analysis showed that, *ceteris paribus*, more HCWs led to an increase in infections, the modeled intervention was to send HCWs home once they became sick. Within the ABM, this was modeled as removing the HCWs from the simulation three hours after they became infected and running short-staffed for 90 minutes until an additional HCW entered the simulation to replace the absent HCW. If an infected HCW

was the last remaining HCW for that particular job, (e.g., triage nurse), then the HCW would wait until a replacement arrives before leaving the simulation. This effectively kept them in the simulation in an infected and infectious state for that period of time. Again, while the infection model was simplistic, it yielded insights into explanatory nature of the ABM approach. It is known, for example, that a person can be infectious before they are symptomatic. Future refinements to the ABM will incorporate finer distinctions between infected, symptomatic, and infectious states.

The OLS regression was used to analyze the effect of these four policies within the given instance context. The dependent variable, as in the previous analyses, was the percentage of patients that acquired an infection between 144 and 168 hours in the ED. Each policy was represented by a dummy variable, was set to one for the simulations where that policy was effective, and set to zero when it was not.

Results for the OLS regression comparing the four policies and policy baseline are shown in Table IV. The R^2 for this model is 0.1142. All variables are highly significant (p-value < 0.001). Among the four policies, dismissing LP patients had the greatest effect. Implementing this policy (compared to implementing no policy) reduced the percentage of patients that became infected by 5.081. Masking infected LP patients reduced the percentage of patients that became infected by 3.159; masking infected HCWs reduced the percentage of patients that became infected by 2.288; and dismissing infected HCWs reduced the percentage of patients that became infected by 2.380.

The results of both analyses suggested that for these particular model parameters, more patients and HCWs in the ED led to a higher percentage of patients getting infected, and that, as a result, the policies that are best suited to reduce the spread of infection are those that most effectively minimize this transmission mechanism. Hence, patient-oriented policies tended to have a larger effect, compared with policies that targeted HCWs. This is further emphasized by the perverse result that increasing the number of registration nurses increased the percentage of patients that became infected, and that increasing the number of doctors, when controlling the number of infected HCW, also led to an increase in the number of infections. It appeared that dealing faster with patients who were waiting in the ED led to an increase in contacts between infectious agents and the primary transmitters (HCWs). These findings were specific to the instance context defined earlier; however, equally important is the general validation of the ABM approach derived through the findings, which develops the foundation for extensions to other instance contexts.

3.7 Discussion

The model and results are reflective of the chosen parameters and the chosen instance context of a representative ED in Winnipeg, Canada. Yet, the work provides insights primarily in two directions. Firstly, through the development of the ABM framework specifically designed to simulate the spread of contact-centric infections such as IV through an ED, the limitations of the model itself are clarified. Simultaneously, the

potential capabilities of the ABM technique overall are illuminated, in that any degree of specificity in topography, agent characteristics, behaviours, and interactions can be accommodated, constrained only by time and computing resources. This ultimately leads to creating a model of extremely high resolution and fidelity. In contrast to other techniques (mathematical modeling), ABM allows users to construct a comprehensive representation of the real world as the available data allow.

Secondly, the work provides insights into the variables that impact the spread of IV infection in an ED, and the concomitant impact of corresponding interventions, as discussed earlier. The research was guided by ED information systems data and available literature. For qualitative validation and verification, the ABM incorporates an animation, allowing the user to view the simulation while it is running. This animation assists in model development as well as providing a decision-maker with an overall view of the ED model while running. The data collected and the analyses performed allowed baseline operations to be simulated, and infection control policies to be introduced and assessed for impact. As such, the ABM is a decision support tool available to healthcare practitioners and policy makers in order to qualitatively assess the relative impact of various infection control strategies, such as early and mandatory vaccination for HCWs, or alternate treatment streams for ILI-symptomatic patients. The ABM illuminates scenarios worthy of further investigation as well as counter-intuitive findings, such as the impact of number of HCWs on new infections. In this regard, it is worthwhile to further consider the research that aims to assess control strategies for nosocomial infections in

hospitals; some researchers have concluded that chance effects and naturally occurring large fluctuations in prevalence can confound the effects of interventions [55],[56], and no model is yet sensitive enough to quantitatively and exactly validate the full range of infection dynamics. Nonetheless, the ABM technique, already validated for urban- and community-level modeling, also holds significant promise for this type of institution-level investigation. Since the agent representation is constant across these various scales of modeling, ABM also offers the potential to integrate different levels of models with one another.

Future work with the current ABM framework will be directed to further fine-tuning the model and augmenting with data obtained more recently from the EDIS of the Winnipeg Regional Health Authority. Comprising seven facilities and approximately 180,000 ED visits over a nine-month period, the data are being used to extract distributions of patient arrival rates, patient triage scores, patient age, patient lengths of stay, and patient waiting times. These distributions will be used to fine-tune the parameters of the ABM framework, in order to enhance the validity of the output. Beyond validating the input parameters, EDIS data will also naturally lend themselves to using the ABM for finer-grained investigations of the effects of infection-control policies, for example, by time of day.

Although still in an early stage, a further extension of this research is to integrate machine-learning technologies within the ABM, where machine learning has potential extensions to decision support or policy evaluation tools. One such machine-learning

technology is genetic programming (GP) [57], with well-documented performance on a variety of complex, real-world problems such as this, and can be implemented effectively with comparatively little expert domain knowledge [58]. One use case would be data mining in order to identify significant inputs. Another might be evolution and optimization of various ED policies, furthering ABM utility as a decision support tool for healthcare policy management. This related work represents the first instance of ABM-GP integration towards this particular problem.

3.8 Summary

This chapter presented an ABM developed to investigate the spread dynamics and intervention policies of nosocomial infections in a hospital ED. This work was collaborative in nature with the primary development of the ABM attributed to the author as well as the data collection for statistical analysis. The author, being most familiar with the ABM itself, also participated in determining types of data best suited and retrievable for analysis, and types of policies that could be simulated within the context of an ABM.

4 ABM 2: RFID Patient Tracking in Healthcare

4.1 Introduction

This chapter presents additional utility of ABMs when applied to further extending the previously developed ABM to include an RTLS. This work was also a collaborative project with the majority of the ABM development being attributed to the author. The purpose of this chapter is to demonstrate another example of an ABM for a healthcare application with a focus on system error, and in doing so, to provide insights into the strengths and weakness of the underlying ABM engine. A version of this chapter is published [47].

Increasingly, healthcare management systems include investment in, and implementation of, technology to track the status and movement of various entities within the healthcare environment. This includes patients, HCWs and physical assets. Such systems are often used as a means of understanding patient flow, controlling inventory, tracking equipment usage, and thereby (ideally) assessing efficiencies in order to optimize resources and processes within that environment [59]. The focus of this chapter is to explore the potential of RFID systems modeling within ABM, particularly in relation to an understanding of the nature and extent of system error that is often overlooked. The healthcare environment was chosen as it is an increasingly complex and interesting application area for RFID, and in which a wide range of RFID-based applications and

devices already exist and can be envisioned for the future. The insights gained through modeling provide a complementary set of data to those gained from the knowledge of performance in existing installations. To that end, this chapter focuses on a case study of an ABM of a hospital ED, with extensions to modeling the provisioning of a RTLS using RFID for patient tracking.

4.2 RFID Healthcare Applications and Patient Tracking Systems

The current scope of RFID technology in healthcare includes conventional, as well as emerging 'smart' RFID devices and applications. The overall objective of RFID applications in healthcare is to automate manual processes as well as to reduce the time required to track and locate assets; this includes vital supplies and people, and thereby results in institutional and productivity gains [60]. Ultimately, RFID applications in healthcare can support service providers in decision-making functions and arduous tasks typical within a complex clinical environment. Over the range of applications, RFID in healthcare generally shows capacity to:

- identify, track, and locate patients, healthcare personnel, medical files, equipment, and supplies, medications, and laboratory and pathology samples;
- detect incorrectly packaged medications or supplies and doses;
- implement systems to recall counterfeit or contaminated medications and supplies;

- correctly identify and locate potentially ‘misplaced’ surgical instrumentation, even those accidentally left within a patient during a surgical procedure;
- confirm medical supply and surgical instrumentation sterilization;
- insure safe and proper handling and disposing of bio-hazardous materials, expended medications, medical supplies, devices, and waste; and,
- even validate actual anatomical locations for surgical procedures. Thus, at the patient point-of-care, RFID can serve to correctly identify patients to receive, and care providers authorized to deliver, a prescribed medical treatment or procedure. Overall, in the healthcare sector there are many emerging RFID inventions that can offer drastic improvements in processing times associated with medical records retrieval, efficiency of service and reliability of medical diagnosis, dispensing of medications, and personnel management [61],[62]. The overall goal of these technologies is to facilitate improvements to patient safety and quality of care, and to do so in a cost-effective manner.

4.3 ABM of RFID Placement for Patient Tracking in an ED

Evolving RFID RTLSs are meant to augment and/or automate the existing data capture (electronic records systems) in place in many healthcare institutions. A typical patient trajectory capture system may include the collection of time of arrival, time of triage and registration, time and duration of treatment including specialty consultations, and time of discharge from the ED. In some cases, data are entered by healthcare workers for

multiple patients, in aggregate when there is a break in the workflow. As such, there may be considerable uncertainty associated with the data themselves, making it difficult for policy makers to make statistically verifiable decisions as they attempt to optimize patient access and flow through the healthcare facility. In this context, RFID systems offer a complementary and more automated means of data collection. However, critical issues are again associated with uncertainty, requiring assumptions to remove ambiguities in the data.

To preface a discussion on error and uncertainty in RFID systems, a brief discussion of the system hardware is introduced. For this purpose, RFID systems are considered to be tag-and-reader systems, comprised, at minimum, of a number of tags and at least one reader. These RFID systems operate over a variety of frequency ranges and with a variety of complexities associated with the tags as well as readers. In general, a tag is associated with an asset or patient and the reader is associated with a location in the physical environment. The tag can be either passive or active, the latter requiring a battery allowing it to power a transponder and be interrogated by a reader. A passive tag obtains its power from the field of the reader, allowing it to effectively 'transmit' its identification back to the reader [63]. Passive tags are modeled in this chapter.

Several sources of uncertainty and error become immediately evident, although are often overlooked by users. Firstly, once a tag is read, it is primarily a proximity measure, meaning the tag is somewhere within the proximity of that reader. Low frequency, passive tags (< 20 MHz) use inductive coupling as opposed to propagating

electromagnetic radiation, and as such, are typically limited to fairly close proximities of approximately 1–2 metres. As a consequence, when attempting to create a patient trajectory through a healthcare facility, time-stamped historical data are required and rules of inference are used to estimate a patient's approximate location.

Secondly, a particular reader-and-tag system will have time-varying, non-isotropic capture areas, impeded by distance and other objects in the surrounding area. Thirdly, in the case of the passive tag with virtually no processing power, a reader may read a tag at apparently random intervals or may not read the tag at all, depending on tag orientation and its environment. A fourth issue is interference, where closely-spaced readers interfere to the point where both readers fail to read a tag that passes through or stays within the interference zone. The problematic case is the lack of a read (missed read) by one or both readers. A study estimating service queue lengths using RFID [64] highlighted the extent of uncertainty inherent even in a straightforward RFID RTLS application.

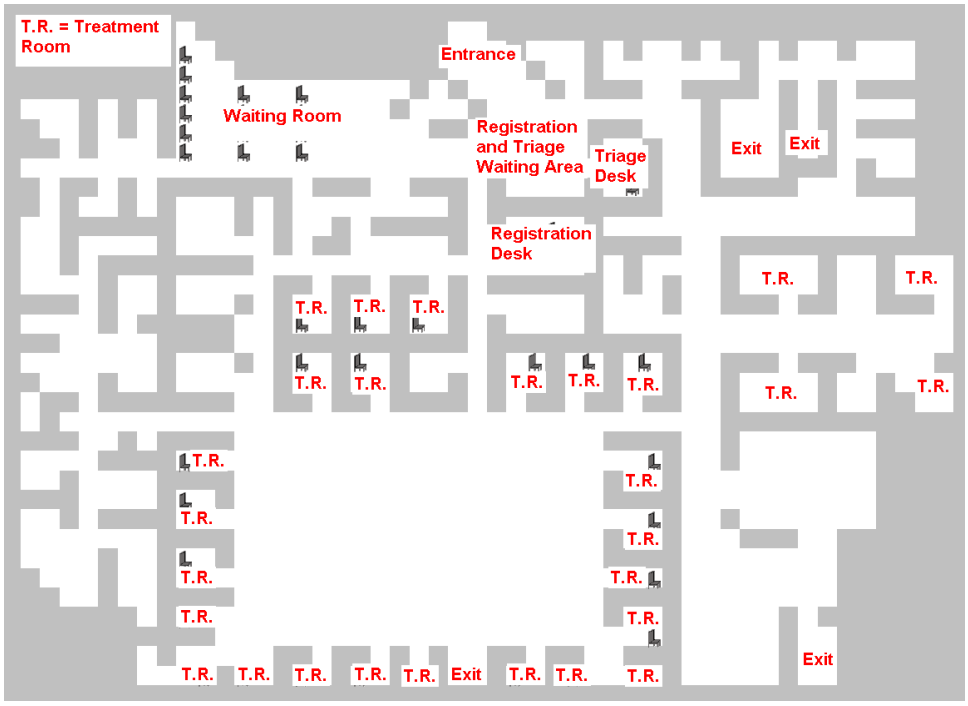
While both passive and active RFID tracking technology will continue to evolve, and uncertainties in operation and quality will be mitigated, these uncertainties will likely never be completely eliminated. As such, modeling plays an important role in optimizing the implementation of any RFID RTLS patient tracking system.

Many ABMs are developed to gain a better understanding of operations through the use of 'what-if' scenarios. An ABM initially developed for improving patient access to healthcare [43], was extended here to allow modeling a RFID RTLS augmented ED, with the objective to gain insights into the nature of error and uncertainty associated with

patient tracking. Of primary interest to a RFID RTLS was the accuracy and precision of patient trajectory through the ED. As each agent behaved in an autonomous fashion under the control of the simulator, an exact trajectory was recorded for each patient, which was then available for comparison and validation to trajectory data captured via the RFID RTLS model.

The basic model allowed for various configurations, including provisioning the number of HCWs, the number and characteristics of patients, as well as the topography of the ED. Operational parameters in the ED included registration, triage, waiting and treatment areas. The ED floor plan used here closely resembled that of a large, acute-care, metropolitan hospital (Health Sciences Centre in Winnipeg, Manitoba, Canada), although tailoring of floor plans is possible. A schematic of the ABM at an instance of time is illustrated in Fig. 9. Patients of varying levels of acuity and HCWs were illustrated within the ED with different. This representation also provided a visual animation as the simulation progresses. In effect, the animation was a tool used to transfer information between practitioners and modelers, whereas the extensive collection of data for statistical analysis proceeded without any type of visualization.

(a)



(b)

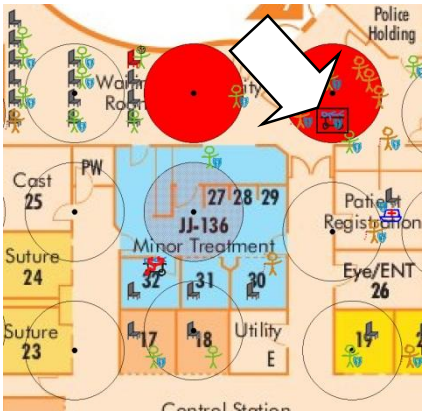
Figure 9 a) visual ED layout, b) logical ED layout of (a)

The RFID RTLS environment allowed for the overlaying or placing of RFID readers, which served as inanimate agents within the ABM. Reader placement can be engineered, or simply distributed in a somewhat systematic manner. For the purposes of this type of simulation, the latter was used to allow for placement roughly based on the density of readers with a specified granularity. In practice, actual reader placement would be subject to access, mounting, and functional constraints, with read patterns also modified as a consequence of the local environment (walls, furniture, equipment, required clearances, as well as people, the latter being time-varying). Tag collision was not considered in this simulation; it was assumed that over the duration tags are within a reader's range and any colliding or interfering tags would be resolved in a time period considerably shorter than that of the tags traversing the reader. The resolution of the collision would be a consequence of multiple reads of the close proximity tags. This would also not be an issue with more sophisticated tag technology with collision avoidance (e.g., active tags).

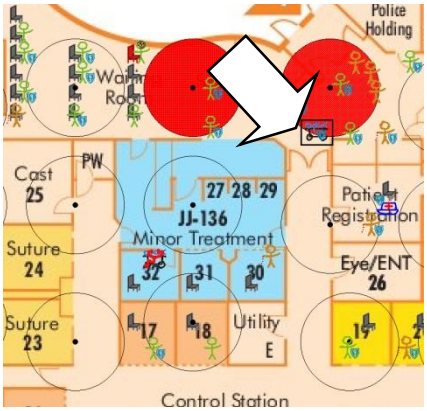
For the RFID RTLS ABM simulation, the exact trajectories of patients were compared with estimated trajectories inferred from RFID tag reads. Patient trajectories through the ED were largely governed by the triage score, by HCW resources, as well as by issues such as social distancing as aspects of agent behaviour. Several layout scenarios were considered: under-provisioning, intermediate-level provisioning, and over-provisioning of readers throughout the ED. The primary metric used for determining the quality of the RFID placement was the spatial error in the patients' actual position as well as the temporal error (duration of time when the patient was outside of a reader's range).

Economic cost considerations have not been included, although one assumes the cost and maintenance would increase monotonically with an increased number of readers.

Figure 10 illustrates the model running with a placement of readers, highlighting a read of a patient tag as the patient traversed the ED. In Fig. 10, patients moved between registration, triage, waiting and treatment rooms in a hospital ED. The reader placement was seen as a coarse grid of concentric dots and circles, with the radius indicating the reader range. The darkest colored circles indicate that a reader has just read a tag, whereas, the lighter colored circles indicate that a tag was read some time previously. Simulation parameters are set to resemble actual records of wait times for EDs of similar resource (number of HCWs, capacity). Although not exact, patient arrival rates and service times were adjusted to reflect times (several hours) spent in ED waiting areas and treatment rooms.



(a)



(b)

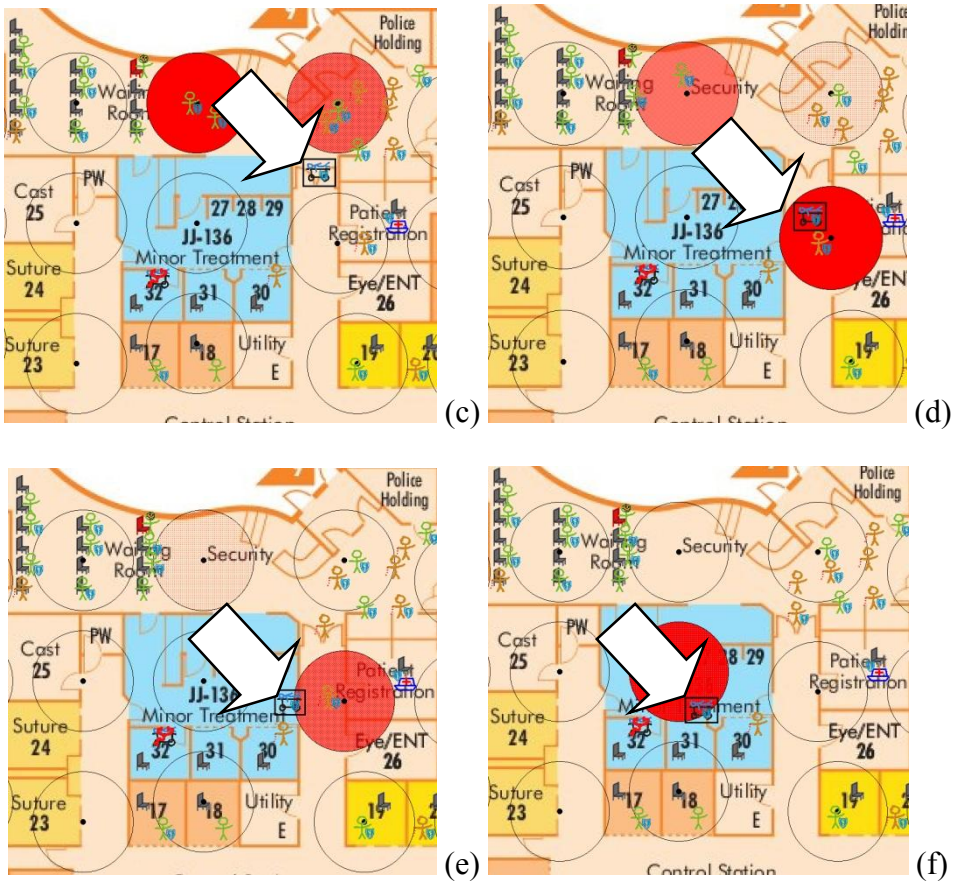


Figure 10. Patient trajectory at time: a) t_1, b) t_2, c) t_3, d) t_4, e) t_5, f) t_6

4.4 Implementation Details

As this ABM instance is an evolution of erSim, the following discussion will be oriented to implementation not previously discussed and/or more directly associated with the ABM for RTLS modeling. Figure 11 illustrates an object inheritance diagram that closely resembles the model used in this study. The RFID RTLS model builds upon an ABM that was initially developed for studying patient access (wait times) and was

subsequently extended for application in modeling the spread of nosocomial or hospital acquired infections. The bold text identifies the extensions added for modeling the performance of RFID RTLS in an ED. In addition to the RFID reader class, modifications were also made to support the data collection in terms of the error model.

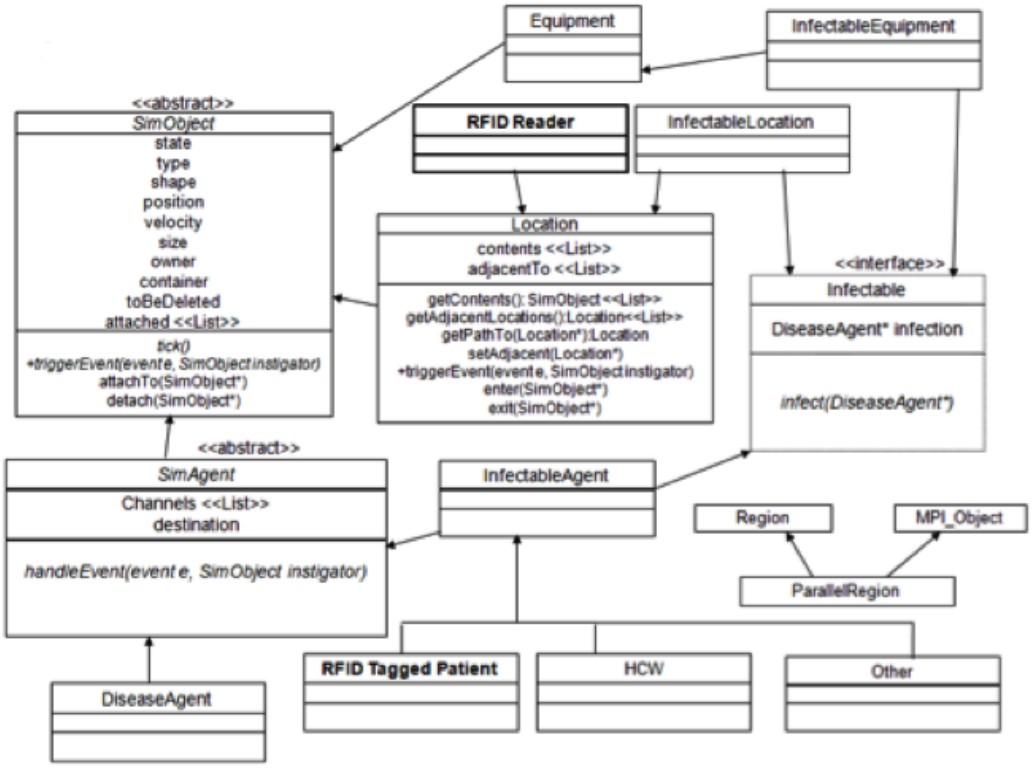


Figure 11. ABM inheritance diagram with RFID extensions

Each agent acted in a set, predictable, but arbitrary order. That is, agents and all other objects were simulated in the order in which they were added to the simulation governed by the arrival rates of patients of various triage scores. Time was discrete, with each time

step having a resolution of 1 second. Patients were modeled as occupying a circular space with a radius of 60 cm, representative of their physical person as well as a concept of personal space.

Using this technology, a hospital ED was modeled as a collection of interacting agents. Patients arrived and required treatment, and were tagged with an RFID tag upon entering the ED. Nurses, who decided treatment paths for patients, directed patients through the system; doctor agents treated patients. The RFID readers were implemented in a similar way to agents, as a class, which read tags of every patient that entered their reading radius.

Agents practice rudimentary path planning, typically choosing the shortest path to the destination location (e.g., travel from waiting room to treatment room). The agents also selectively practiced microsocial-distancing, where they tended to distance themselves according to their local density. These are examples of localized agent decision-making.

A simulated RFID reader noted when an agent (patient) entered its reading range, a roughly circular region with occlusions primarily due to walls. The RFID model required a line of sight between the reader and tag; walls provided 100% attenuation in the simulation. The model can be modified to allow for propagation through walls, but would require an attenuation model to account for a reduced range in the previously occluded region. Furthermore, interference regions were modeled where multiple readers' ranges overlapped. A placement algorithm was employed, which attempted to equally position readers in a grid throughout the modeled ED floor plan. The ABM also allowed for

readers to be placed manually, as would be the case in an actual ED with more detailed planning. A heuristic placed the reader on either side of a wall if the equally spaced grid placement algorithm should place the reader inside a wall; this was referred to as a naive grid placement strategy. Long-term goals of an ABM would be to further develop the feedback loop of an optimizer adjusting reader locations in a manner to minimize uncertainty. In doing so, one would also have to model the actual radio frequency environment via a site survey, and model the performance of the system taking into account interactions in a system with a plurality of patients, visitors, HCWs, as well as portable equipment.

No signal or notification was received when the tag left a reader's reading radius. Furthermore, better resolution than the error resulting from the reader's own radius was not expected. That is, a patient that entered the reading range of a reader was known or thought to have remained within that particular reader's range until another reader event was triggered. In general, a tag may be read multiple times while within a reader's range. These multiple reads in effect confirm the patient's proximity and were implicitly accounted for here. This is an optimistic error model as when an agent is within a read radius, the contribution to spatial as well as temporal error is zero.

Since ABM is very compute intensive, parallelism should be used wherever possible. Parallelism was implemented during the collection of data, as the ABM was tailor-made to not only exploit fine-grained parallelism, but also exploitable at the coarse-grain level.

The latter parallelism was exploited here where individual ABMs would executed on a small compute cluster.

4.5 RFID Patient Tracking ABM Simulation Results

4.5.1 Variable Reader Configurations

The simulation parameters are summarized as follows:

- The ED layout corresponded to an area of approximately 30 by 40 metres (1464 by 2001 pixels on screen). While the scaling is not precise, it was chosen to be a reasonable representation.
- The RFID reader range was chosen as a circular region, with a radius of approximately 2 metres (100 pixels on screen), roughly corresponding to the reading radius of many newer passive readers.
- RFID reader spacing was varied from 2 metres (100 pixels) to 7 metres (350 pixels) between readers, in increments of 1 metre (50 pixels). This created six reader configurations shown in Fig. 12. Readers are indicated by black dots, with their ranges indicated by circles. The placement scenarios are not optimal but are an attempt to capture patient flows through major traffic routes in the ED modeled.
- At each of the six reader configurations, the simulation was run 500 times, with roughly 200 patients visiting the ED in each simulation.

- The variable of interest in the simulations was uncertainty or error, defined as both spatial error and temporal error. The measure of spatial error used in this model is the difference in trajectory inferred from the readers, relative to a patient's actual position as known within the simulation. As such, when a person is recorded within the range of a reader, the spatial error would be zero. As the patient moves beyond the range of the reader, the spatial error is the Euclidean distance from the reader to the actual patient position. The temporal error in this model is defined as the percentage of time that a patient is not within the range of a reader.

Other measures of error, such as Manhattan distances would be equally reasonable and well within the modeling alternatives. Once a patient moves within the range of another reader, both the instantaneous spatial and temporal error would return to zero. A complication occurred when modeling interfering readers (an over-provisioned scenario). Here a dead zone was modeled, reflecting the overlap of two readers' ranges. The difference to the error measure was that the read might be delayed as an agent entered a reader range, which was reduced by a dead zone. This contributed to a longer period of proximity uncertainty or error.


NCE
Family Room
Waiting Room
Security
Police Holding
Psychiatry Interview Rooms
PW
Psychiatry Workroom
STAIRS
ELEV
Pens
Ambulance/ Police Office
Triage
Cast
25
Patient Registration
Medication
A
B
27 28 29
Minor Treatment
Suture
24
Eye/ENT
26
Isolation
22
JJ-171
Resuscitation Room
Suture
23
Utility
E
D
Image Viewing
C
Administrative Offices
Control Station
SW
Coats
JJ-179
TRAUMA OR
Restricted Area
JJ-115
Nursing Station
Clean Supply
JJ-119
Nursing Station
Staff Team Room
Student and Resident Workroom
Image Viewing
RT Blood Gas Lab
Scrub
General Care
Urgent Care
Gyne Forensic
DIAGNOSTI
IMAGING

(a)


NCE
Family Room
Waiting Room
Security
Police Holding
Psychiatry Interview Rooms
PW
Psychiatry Workroom
STAIRS
ELEV
Pens
Ambulance/ Police Office
Triage
Cast
25
Patient Registration
Medication
A
B
27 28 29
Minor Treatment
Suture
24
Eye/ENT
26
Isolation
22
JJ-171
Resuscitation Room
Suture
23
Utility
E
D
Image Viewing
C
Administrative Offices
Control Station
SW
Coats
JJ-179
TRAUMA OR
Restricted Area
JJ-115
Nursing Station
Clean Supply
JJ-119
Nursing Station
Staff Team Room
Student and Resident Workroom
Image Viewing
RT Blood Gas Lab
Scrub
General Care
Urgent Care
Gyne Forensic
DIAGNOSTI
IMAGING

(b)


NCE
Family Room
Waiting Room
Security
Police Holding
Psychiatry Interview Rooms
PW
Psychiatry Workroom
DECON
STAIRS
ELEV 1
Pens
Ambulance/ Police Office
Triage
Cast 25
Patient Registration
Medication
Suture 24
JJ-136 Minor Treatment
Eye/ENT 26
Isolation 22
JJ-171 Resuscitation Room
Suture 23
Utility E
Image Viewing
Administrative Offices
Control Station
Coats
JJ-115 Nursing Station
Clean Supply
JJ-119 Nursing Station
JJ-179 TRAUMA OR Restricted Area
Staff Team Room
Student and Resident Workroom
RT Blood Gas Lab
Scrub
General Care
Urgent Care
ELEV 4
DIAGNOSTI IMAGING

(c)


NCE
Family Room
Waiting Room
Security
Police Holding
Psychiatry Interview Rooms
PW
Psychiatry Workroom
DECON
STAIRS
ELEV 1
Pens
Ambulance/ Police Office
Triage
Cast 25
Patient Registration
Medication
Suture 24
JJ-136 Minor Treatment
Eye/ENT 26
Isolation 22
JJ-171 Resuscitation Room
Suture 23
Utility E
Image Viewing
Administrative Offices
Control Station
Coats
JJ-115 Nursing Station
Clean Supply
JJ-119 Nursing Station
JJ-179 TRAUMA OR Restricted Area
Staff Team Room
Student and Resident Workroom
RT Blood Gas Lab
Scrub
General Care
Urgent Care
ELEV 4
DIAGNOSTI IMAGING

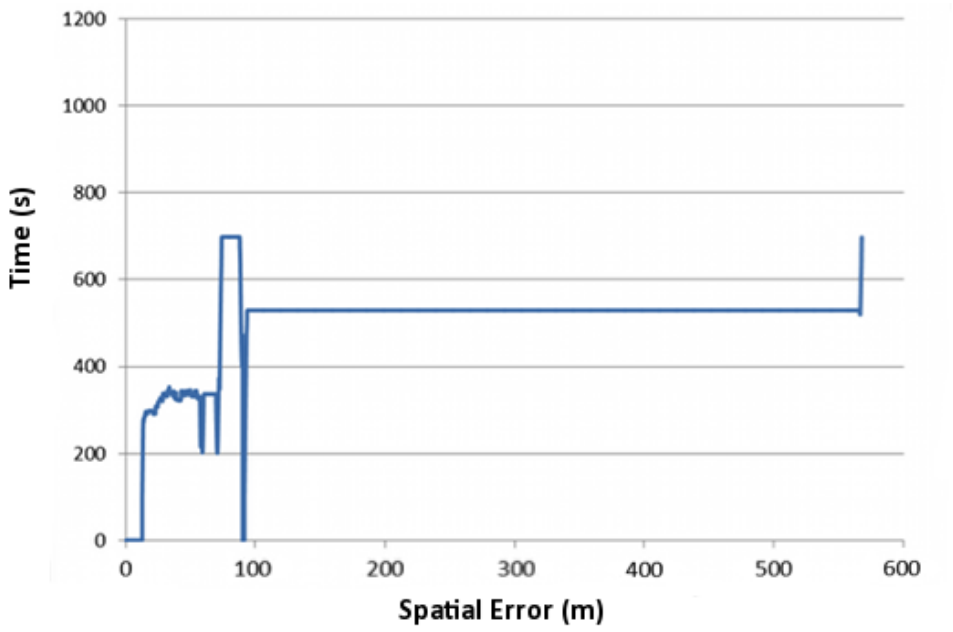
(d)

(e)

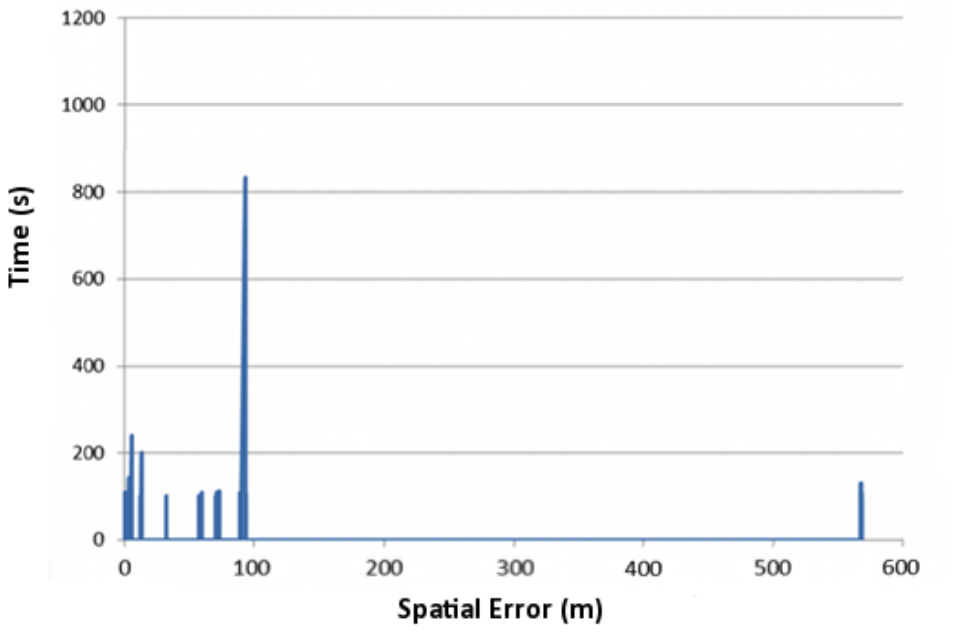
(f)

Figure 12. RFID reader layout at a reader spacing of: a) 2 m, b) 3 m, c) 4 m, d) 5 m, e) 6 m, f) 7 m

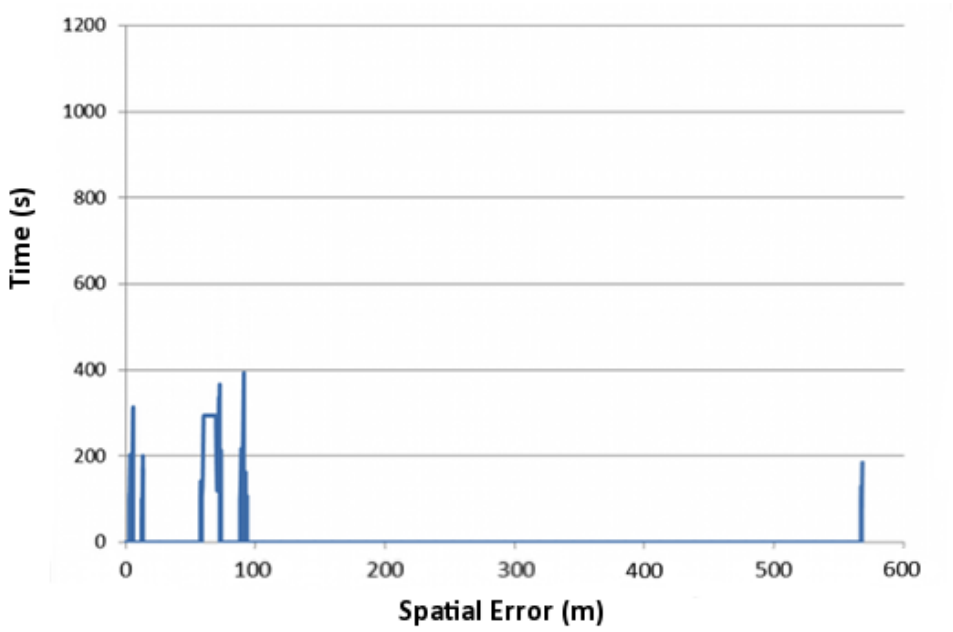
Figure 13 illustrates the typical spatial error between reader-inferred location and actual location for a single patient, plotted over the duration of stay in the ED, for the six reader configurations. The trajectory error illustrated is for the reader arrangements seen in Fig. 12 and for typical patient instances. The behaviour of the patient is highly stochastic, governed by a number of random variables. In Fig. 13, the spatial error is plotted as being the Euclidean difference between the last read reader and the patient location once outside the reader range. When a patient is within a reader's range, the error is set to zero.


Time (s)
1200
1000
800
600
400
200
0
0
100
200
300
400
500
600
Spatial Error (m)

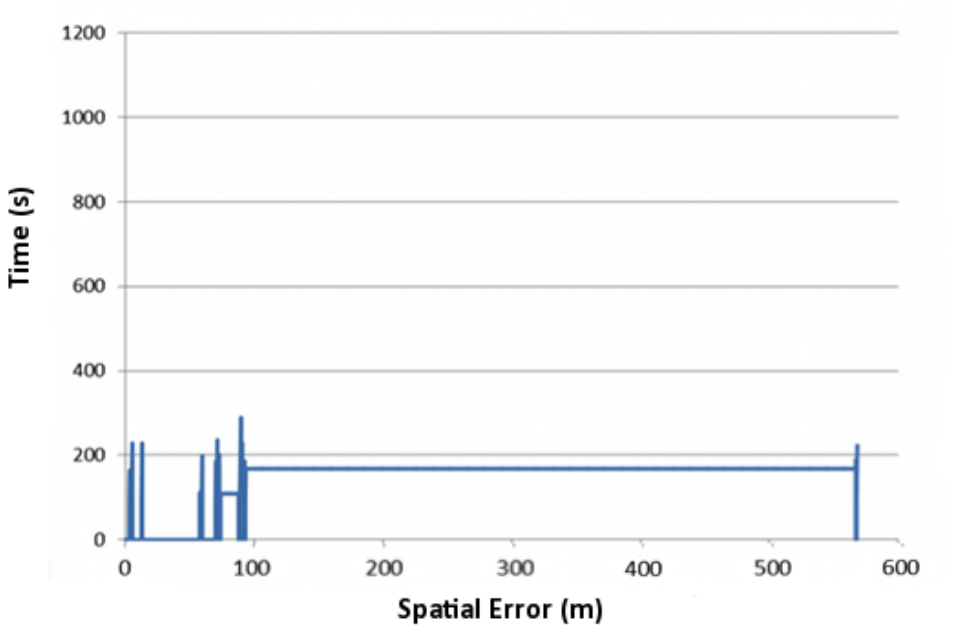
(a)


Time (s)
1200
1000
800
600
400
200
0
0
100
200
300
400
500
600
Spatial Error (m)

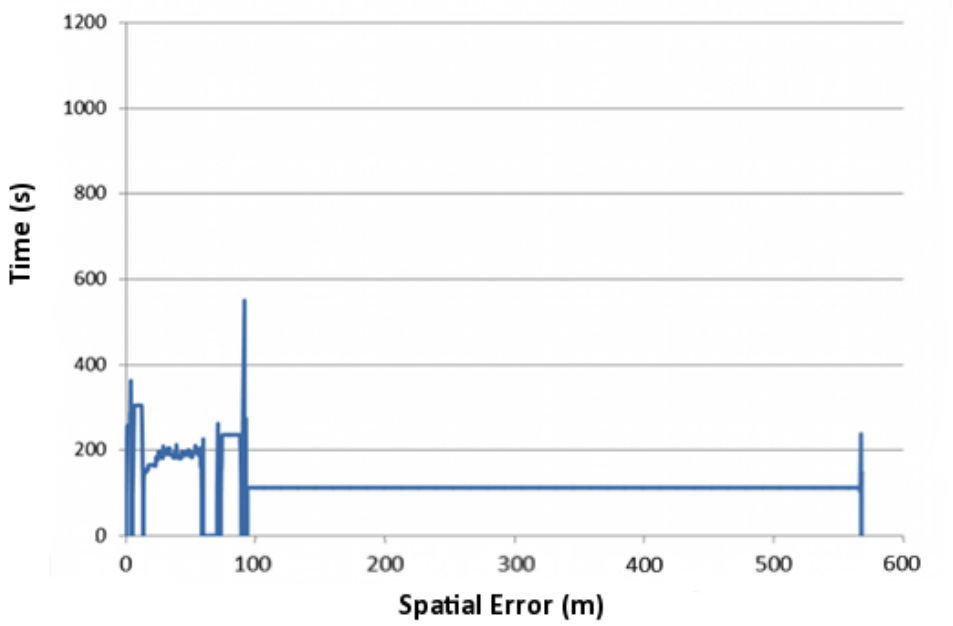
(b)


1200
1000
800
600
400
200
0
Time (s)
0
100
200
300
400
500
600
Spatial Error (m)

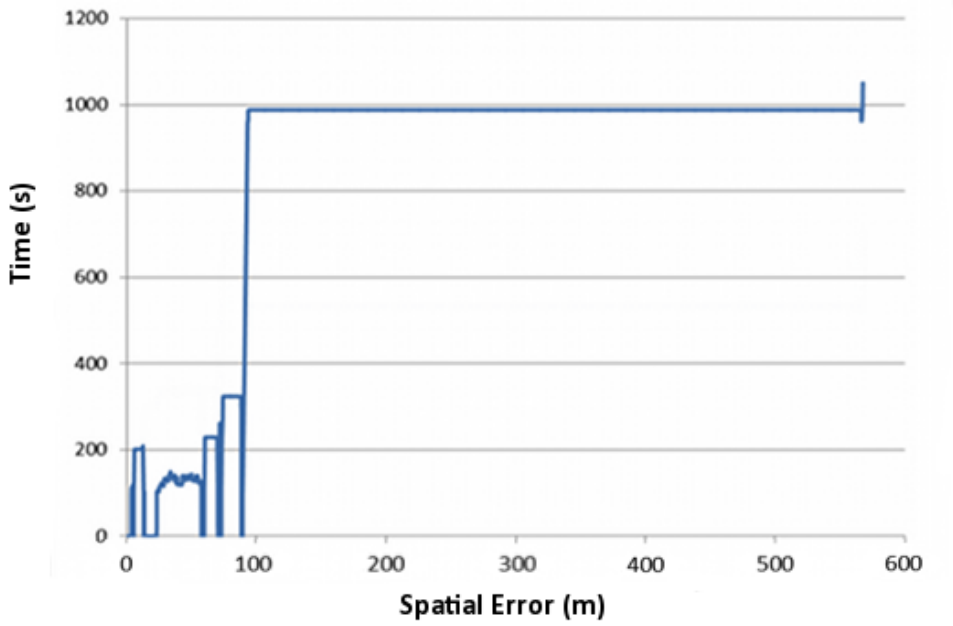
(c)


1200
1000
800
600
400
200
0
Time (s)
0
100
200
300
400
500
600
Spatial Error (m)

(d)



(e)



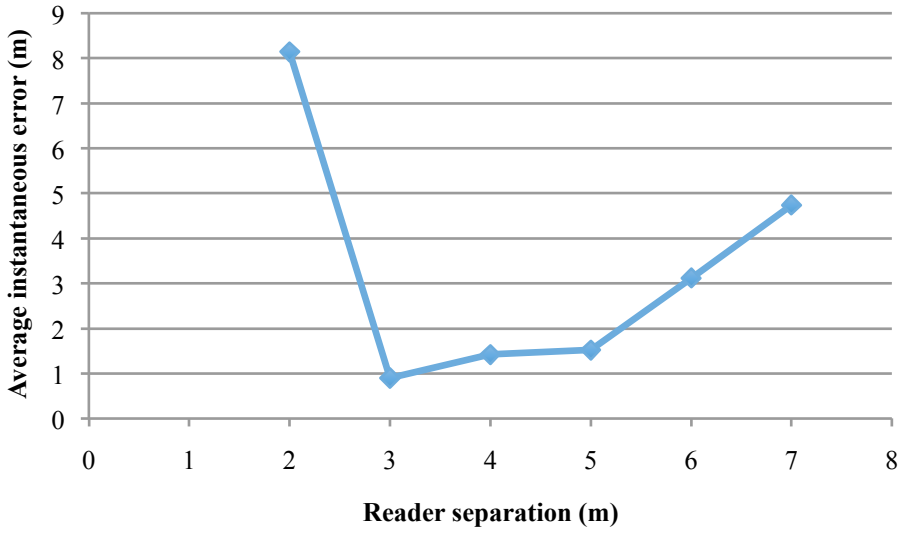
(f)

Figure 13. Spatial error for same patient at a reader spacing of: a) 2 m, b) 3 m, c) 4 m, d) 5 m, e) 6 m, f) 7 m

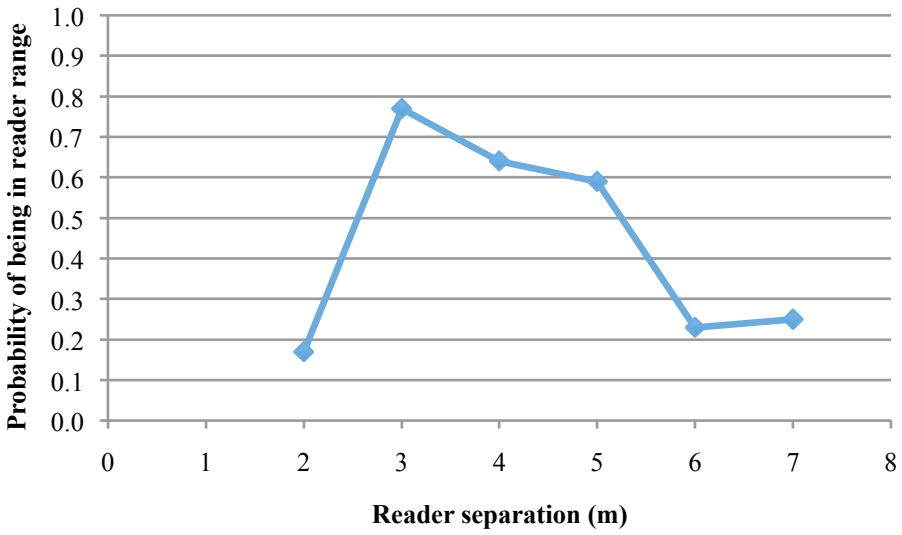
Table V and Fig. 14 summarize the normalized spatial and temporal error. For example, in Table V, a reader spacing of 7 metres, the patient would be in the read range of a reader 25% of the time with a normalized error of approximately 5 metres. As the number of readers increased, the error or uncertainty is clearly reduced. The exception seen in Figs. 13 and 14 is the case of the reader configuration at 2 metres, i.e., closely spaced readers with significant interference zones. Once a patient has been read by a reader, the patient is not considered to be in error if the patient moves out of the active read range and moves through or stays (waits) in an interference zone of that particular reader.

TABLE V SUMMARY OF RFID PERFORMANCE AT VARIOUS SPACINGS

Reader separation (m)	Probability of being in reader range	Variance of probability	Normalized spatial error (m)	Variance of spatial error (SD)
2 (dense)	0.17 (17%)	0.08	8.14	763.98 (27.6)
3	0.77	0.10	0.90	124.32 (11.1)
4	0.64	0.11	1.42	135.74 (11.7)
5	0.59	0.12	1.52	124.50 (11.2)
6	0.23	0.08	3.12	138.76 (11.8)
7 (sparse)	0.25	0.09	4.74	621.98 (24.9)



(a)



(b)

Figure 14. RFID location errors: a) spatial, b) temporal

If a patient moves into an interference zone of a reader and stays (waits) there, the patient is considered to be in error until moving into the active read range of that reader

and being read. One of the most interesting aspects of the simulation was the percentage of time that a patient would be in range of a reader. From these simulations, in an under-provisioned system, the patient's position was only known approximately 25% of the time, whereas, for the over-provisioned case the patient's position was also known approximately 20% of the time. This highlights again the importance of strategic installation locations for the readers (e.g., in the major traffic flow patterns through the topography). Depending on the RFID model, there likely is a near optimal configuration of readers that will minimize the uncertainty associated with an RFID RTLS.

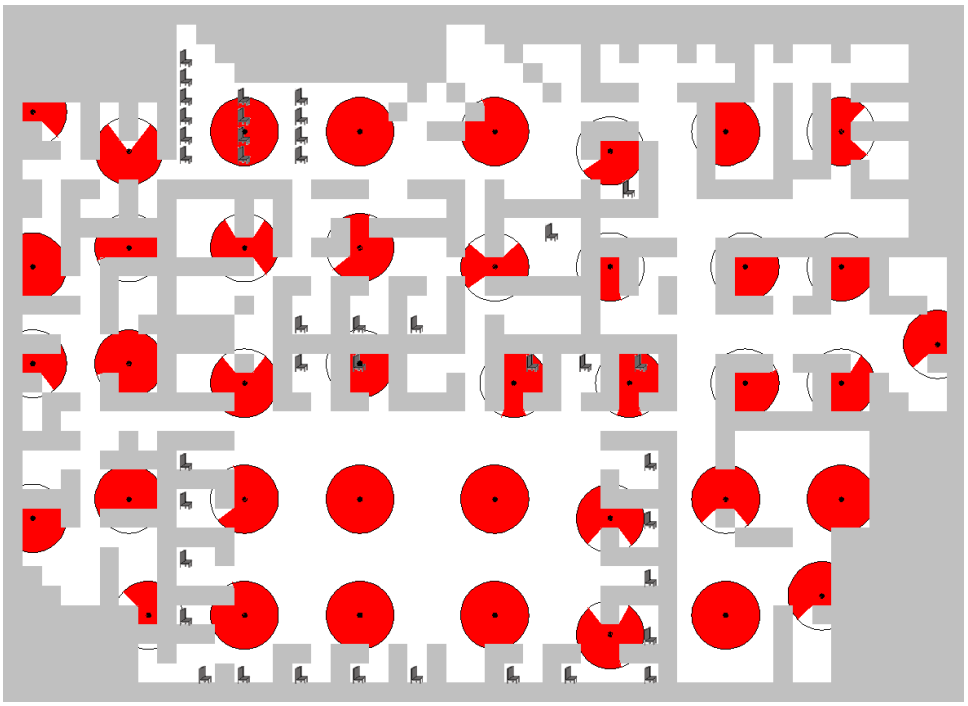
It should be noted that although the uncertainty is generally decreased with an increasing number of readers, there is a point of diminishing return. Although not linear, the cost of provisioning and maintaining the ED with readers separations of 3 metres is significantly greater than an ED with the readers separations of 5 metres. In addition, any inference of behaviour as a consequence of the location of the tag being read was not considered. For example, if the last read reader was located in the waiting room, one can likely infer that the patient is still in the waiting area until another reader reads the patient's tag.

4.5.2 Variable Reader Ranges

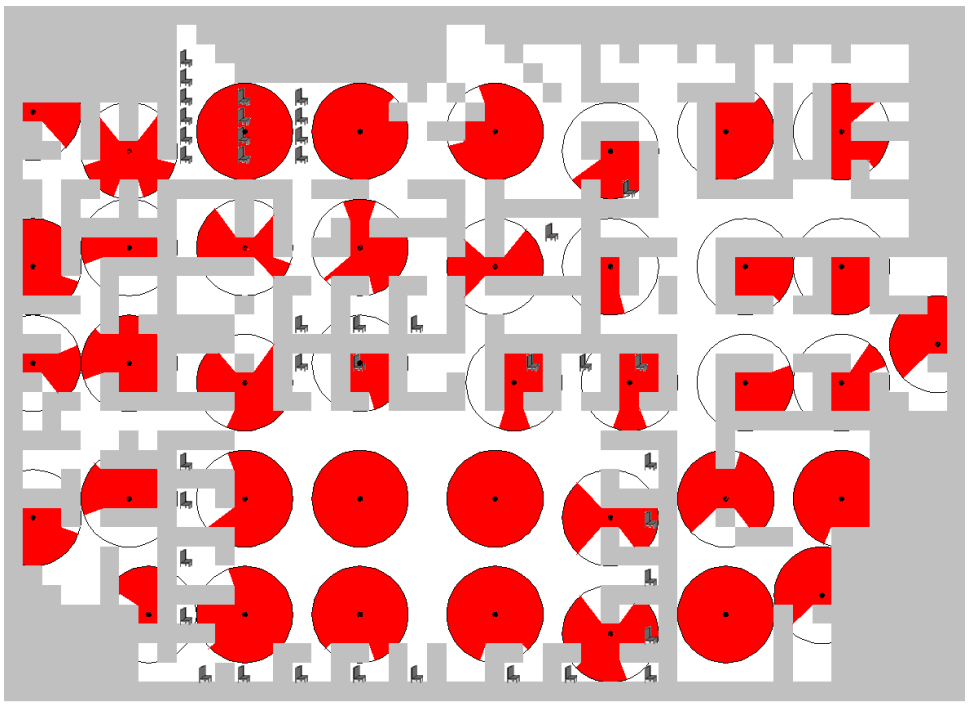
This case study further examined the performance of the readers in terms of their read radius. Readers and associated technologies tend to improve the range of RFID systems over time, whether passive or active. In this simulation, the readers' read areas are either

decreased of increased by a factor of two from the baseline configurations outlined earlier.

The best-case scenario from the previous placement of readers was denoted when they were spaced 5 metres apart. This scenario was argued as function of reader error and diminishing return if more readers were deployed (Fig. 14). Based on other metrics (Fig. 13), one could also argue that the reader configuration at a spacing of 4 metres is a best-case scenario. As is typical, implementation-specific priorities and their associated metrics (cost, performance, etc.) define a best-case scenario for a given user. This section illustrates other variations of reader capability that are also easily modeled within the ABM framework. Figure 15 illustrates the case where the reader range (coverage area) is reduced or increased by a factor of one-half or two, respectively. Figure 15b is the same as the previous case with readers spaced 5 metres apart having a read radius of 2 metres. An identical number of simulations were performed (500) with the results presented in Fig. 16. One can see that the case of readers spaced at 5 metres, with a 1 metre read range is likely close to the ‘near optimal’ configuration. The trade-off again would be diminishing return if one were required to invest in the large coverage readers, as cost tends to increase with read range.



(a)



(b)

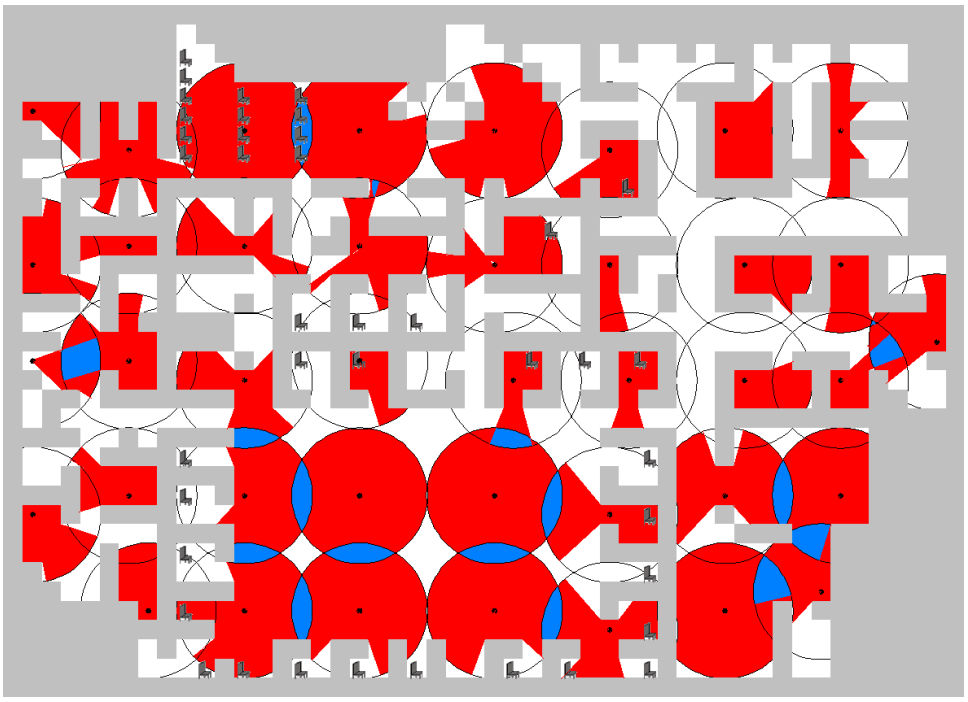
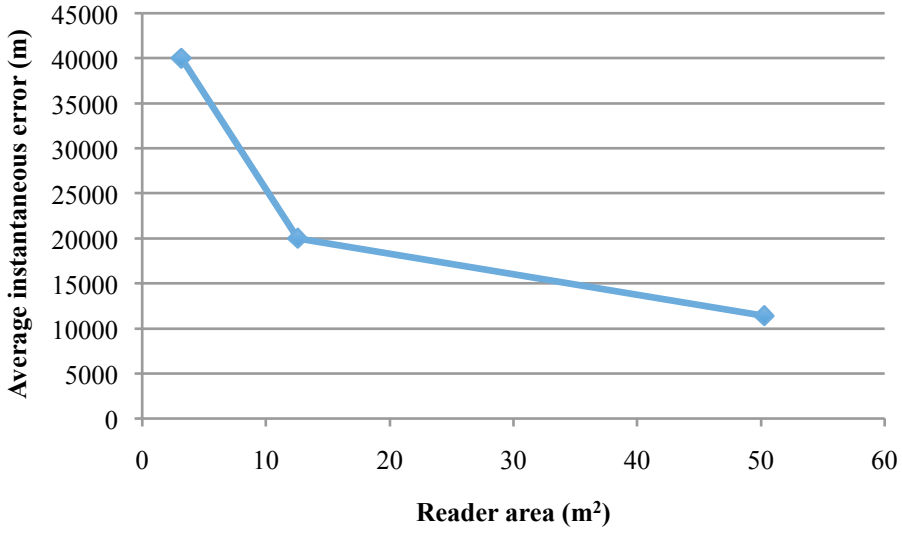
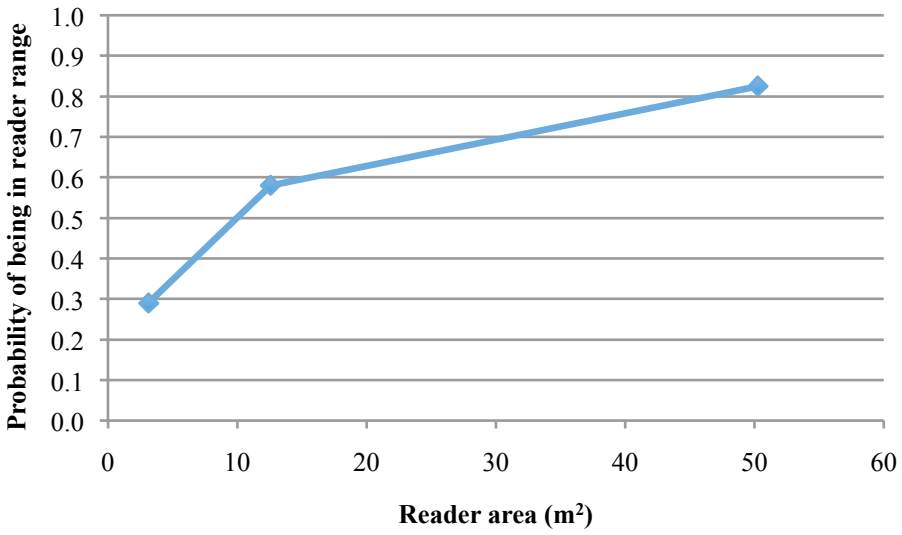

(c)

Figure 15. RFID coverage (red) and interference (blue) areas for reader radii of: a) 1 m, b) 2 m, c) 4 m



(a)

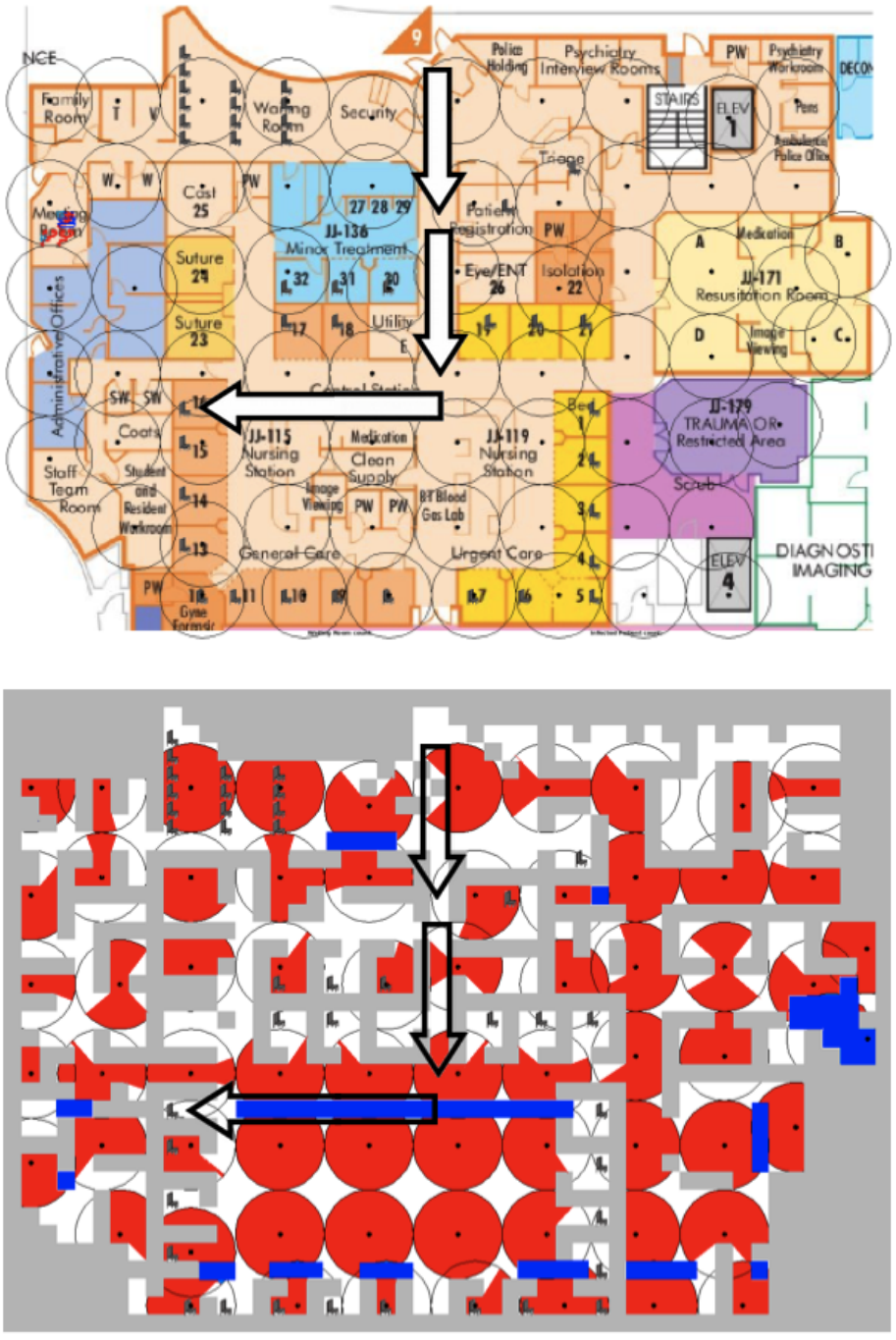


(b)

Figure 16. RFID range performance: a) spatial, b) temporal

4.5.3 Error Interpretation

It is difficult to illustrate explicitly the effect of the various errors (Fig. 16) seen with an arrangement of readers. It is, however, enlightening to attempt to graphically illustrate why some reader placements result in large error in estimating a patient's location. Figure 17 illustrates the coverage areas, blind spots, and regions of modeled interference. Red illustrates coverage area, with blue representing regions of interference (the over-provisioned case). Ideally, the reader would have a circular red read area. Degraded read regions would be a result of walls and obstructions where the tags would not have a clear path to the reader and thus, in the passive case, be unable to acquire sufficient energy to function. Degraded performance also occurs in regions of reader interference where the readers are too closely placed. From this perspective, it is relatively easy to identify poorly performing patient trajectories. For example, from this illustration and observation of patient trajectories, one can envision a patient making their way to a treatment room not equipped with a reader and waiting there for treatment for a considerable period of time. In these cases (which may be pathological), the error can be considerable due to hours of wait time as seen in Fig. 16a. In Fig. 17a, the arrows illustrate a path a patient may traverse to a treatment room. Overlaying the view seen by the ABM (Fig. 17b), one can see that the patient trajectory traverses a region of considerable interference (horizontal arrow), ending at a treatment room without read coverage (occluded due to walls).



(a)

(b)

Figure 17. Trajectory of outlier patient with high temporal and spatial error: a) actual layout, b) logical layout showing coverage (red) and interference (blue) areas

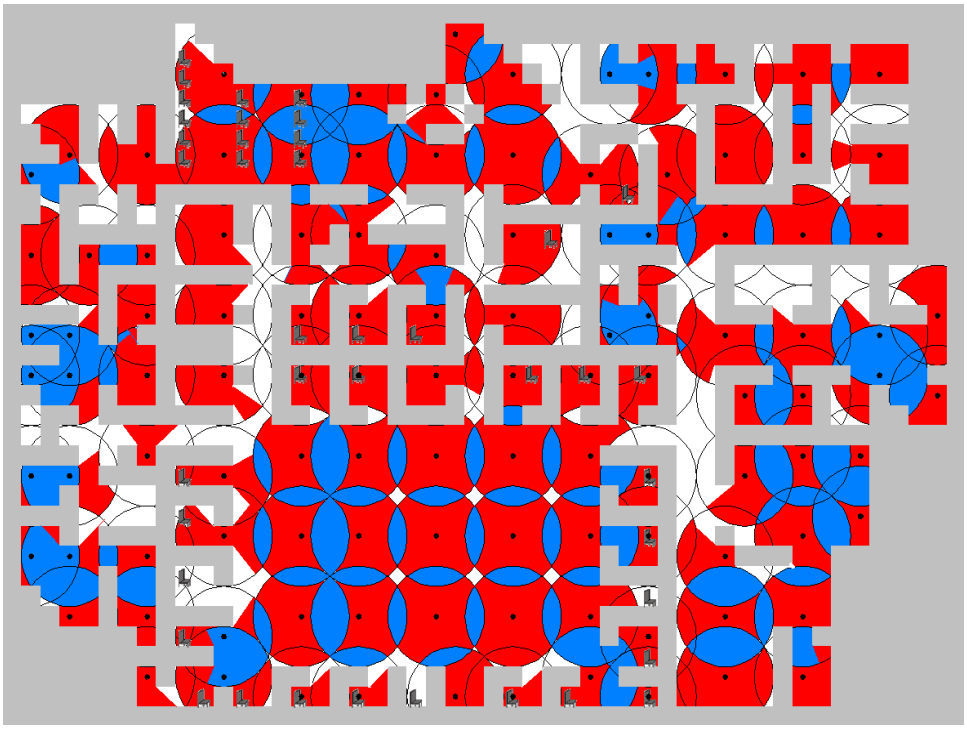


Figure 18. Dense reader layout increasing interference (blue) areas

Figure 18 illustrates the difficulty with simply increasing reader density if interference is a problem. Red illustrates the percentage of floor space covered by readers while blue illustrates regions of interference. This corresponds to the high error seen in Figs. 13a and 14 in the case of reader spacing at 2 metres (over-provisioned), where interference is modeled as error.

There are techniques (beyond the scope of this chapter) for reducing the error associated with estimation of patient trajectories. The simplest are those based on the Kalman algorithm as well as others based on conditional probabilities [65]. Figure 19 illustrates a histogram of a passive tag's maximum read distance from a GAO GP90

passive reader using clamshell tags. The data was collected under ideal conditions and as such represents a bound to error anticipated in the read itself. It does, however, lend credibility to the ABM with fairly abrupt read ranges simulated. Any backend system deployed for collection of RFID RTLS data would presumably be also equipped with estimation software to further reduce error.

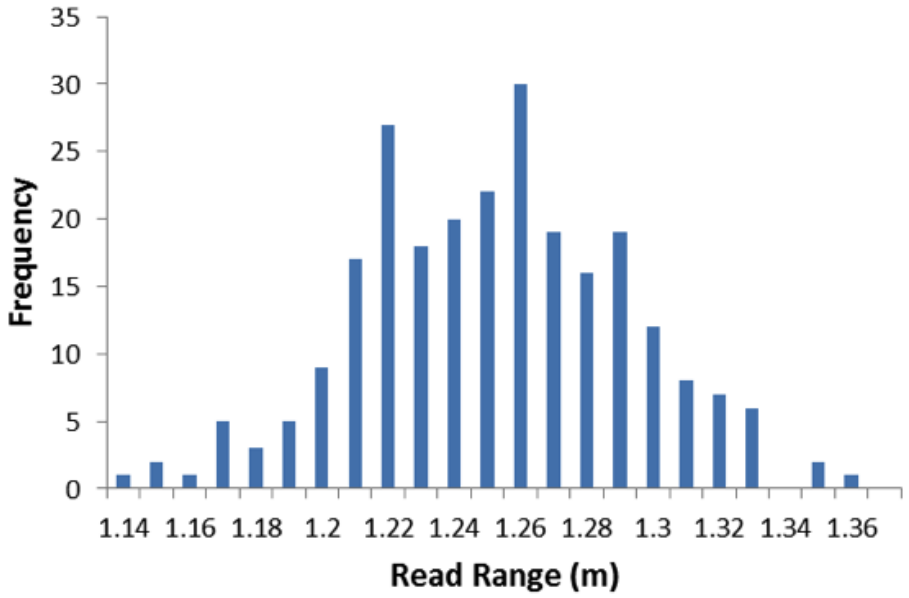


Figure 19. Histogram of GP90 read range

4.6 Summary

Chapter 4 presented extensions to erSim that allowed the modeling of simulated deployments of RFID RTLS. This research was in keeping with the early objectives of,

namely, the illustration of the utility of ABMs for modeling and simulating hospital environments. Again, this work was a collaborative project with the primary contribution of the author being the development of the ABM engine, running simulations, and collecting and analysing the data.

5 ABM 3: Zombie Modeling for Disaster Preparedness

5.1 Introduction

A further application of the ABM research was to investigate disaster preparedness scenarios, and the opportunity arose to address this through an initiative to model zombies using an ABM within a variety of environments. These included zombies on a country-wide scale, an urban scale, a campus scale, and within an institution (e.g., ED). For this undertaking, erSim was naturally redeployed to include a considerable number of agents' behaviour modifications to include an outbreak of zombies in an ED.

The author was responsible for the development of the zombie ABM in an ED. The goal of this work was also to provide insight into how various disaster-preparedness and disaster-response policies may be better evaluated in a simulated environment, as opposed to best effort and expertise or experience. As with previous chapters, the outcome was to provide the author with additional insights for the creation of a more viable ABM framework. Although the chapter includes some levity, it is not without the more serious aspects associated with ABM, that being creating agents with greater diversity as well as facilitating or demonstrating the role ABM plays in policy evaluation.

5.2 Background

Zombies are widely believed to be one of the most interesting of the 'evil' creations, primarily because there is the potential for an epidemic of them. After all, they are the 'living dead' and we will all be dead at some point.

For mathematical modeling purposes, this means that there is a remote possibility, through no fault of one's own, that they could wake up in the morning suffering from a pathogen causing a zombie-like illness (ZLI), which could develop into a full-blown zombie infection. In the case of institutional-level modeling of zombie infection, such as within an ED, this would be recorded at triage as a ZLI. This would be followed by the usual several hour wait in a waiting area, where others waiting would likely practice some degree of social distancing as the disease progresses through the infectious state to a full-blown zombie state.

From a modeling standpoint, it is fortuitous that we have recently undergone a significant H1N1 and SARS pandemic, as much of the mathematical modeling and methods can now be brought to bear on problems of significantly greater deleterious impact. It is widely recognized that we are long overdue for a very significant zombie outbreak, and being unprepared for such disasters will result in bigger disasters.

5.3 Zombie Models

The modeling methodology employed here is again ABM. Although the research carried out here is based on zombies and ZLI, the ABM methodology can be applied to

many real-life illnesses with similar vectors of transmission. In particular, ABMs are well suited to modeling disease spread where the vectors of transmission are by direct (close) and indirect (casual) contact. It may also be the case that disease spread, impacted by more specific behavioural patterns (e.g., sexual orientation, intravenous drug usage), will be amenable to ABM modeling if one is able to source the requisite data and build agents of similar complexity. This chapter includes references to animations of the models on YouTube.

5.3.1 Zombies in the Emergency Department

This section discusses zombies in a hospital ED. This is a very reasonable modeling scenario, as early on in a zombie epidemic, those experiencing ZLIs will likely seek treatment within the healthcare system for their mysterious illness. It also represents a good place for a zombie epidemic to get a foothold, as zombies are notorious for being able to quickly develop like real-world antibiotic-resistant strains, making them resilient to whatever treatment would be prescribed.

In this instance, for zombie modeling and purposes of establishing gross behaviour patterns, a simplified ED floor plan was employed. It is important to note that within ABM, the greater the accuracy of the overall environment modeled, the better the simulations will be. This is somewhat counter to other methods which exploit generalization, whereas, ABMs attempt to exploit specification. This provides advantages as well as disadvantages for ABMs. For example, ABMs can include more accurate

assumptions of disease characteristics and human behaviour and describe population heterogeneities better. On the other hand, large amounts of data are required to specify the model parameters and ABMs can be very computationally expensive. Within the setting of an ED, there are now emerging protocols and technologies for tracking individual patients and their flows or trajectories, thereby lending themselves for use within an ED-ABM framework.

The ED is staffed with HCWs, who are either doctors or nurses with various responsibilities. Patients arrive at rates roughly based on real data and are triaged at various acuity levels. Among these patients are those arriving and presenting with a ZLI, usually feeling lethargic, with headache, and other flu-like symptoms and general malaise. Once triaged, they typically wait in a waiting area for several hours, which could be problematic for other patients, as there is a very real chance to transmitting the zombie infection. The most typical mode of infection transmission is through a bite from a zombie within close contact range, where the pathogen is transmitted by saliva.

Early on in the simulation and while the ED waiting room is not too congested, patients tend to practice some form of social distancing, such as trying to maximize the distance between themselves within the waiting room. Although health agencies are doing the best they can in accommodating patients while they wait, there can be an occasional grab or bite, which results in transmission of the infection. Similar to ILIs, ZLIs are also identified in part by a set of physical symptoms, which trigger behaviours in others.

As with the other ABMs developed here, people are modeled as being in a particular state, during the outbreak. They may either be susceptible, exposed, infected and developing symptoms, or in full catatonic zombie state. In this state, there is no acclimation period; a zombie will begin hunting immediately upon transformation. Of course, in the 'real world' they can also be shot in the head, in which case they die. At this time, there is no acceptable hospital policy allowing an ED physician to kill a patient as treatment.

For much mathematical modeling of diseases, one is interested in the impact of a mitigation strategy. For our purposes here, interest lies in studying both the collapse of the ED during the initial stages of a zombie outbreak, as well as impacts of mitigation policy. Within the modeling parameters, one can adjust the arrival rates of patients presenting ZLI, the transmission probabilities of close and casual contact, and the immunity or lack thereof of the HCWs.

5.3.2 Simulation Scenarios

Within the ED, there were several scenarios simulated. The basic difference to modeling an ED with zombies, as opposed to more traditional patients, was in modifying the behaviour of zombies, traditional patients, HCWs, as well as developing and simulating intervention policies. For patients as well as HCWs, additional behaviours were introduced. These included the abilities to flee, shoot, and shove. These are described briefly below, and in this discussion, uninfected agents are called 'survivors'.

- Flee behaviours: Depending on whether attacked by a zombie, or perhaps triggered by seeing other fleeing survivors, a survivor has the ability to flee. A special instance of ‘flee’ included hiding for 5–10 minutes. In this case, an uninfected person had the ability to hide in a random room. Patients that fled and left the ED never returned, but HCW's were replaced after 90 minutes as per an intervention policy in which infected HCWs were sent home.
- Shooting behaviours: The probability of an agent being ‘immune’ (vaccinated in a traditional sense) was translated as the probability of the agent carrying a shotgun. If armed, there is a probability that an agent with a shotgun could shoot a zombie within a survivor perception range. This distance was a radius larger than the casual contact range, but within shotgun shooting range. There is a cool-down period for the shotgun (meaning an agent needs time to reload and aim at a new target), as well as a probability of hitting the target. Shot zombies are dead (as are any inadvertently shot survivors).
- Shove behaviours: If either unarmed or armed, but still within the shotgun cool-down period and attacked by a zombie, survivors merely pushed the zombies back (probably out of the casual contact range, allowing for a person to flee before becoming infected). This is akin to an ad hoc intervention policy where patients in a waiting room tend to practice social distancing on a micro level.

Combat actions succeeded probabilistically and could have their probability of success parameterized, so there were numerous outcomes for each zombie/survivor encounter.

Combat probability parameters included: successful grabbing, biting, shoving, and shooting. These probabilities are self-explanatory, and were adjustable, allowing for the simulation of various mitigation strategies, various levels of zombie aggression, and patient response.

Survivors maintained a count of the number of zombies that grabbed them, and this resulted in a drag force (scaling down of the survivor's speed) that allowed zombies a better opportunity to bite. Thus, if enough zombies grabbed a survivor, they wouldn't be able to get away. This is a well-known folklore phenomenon as fatigue sets in for survivors battling zombies.

Once infected, and depending on the incubation period (dependent on number of bites), a person becomes zombie, at which time their behaviour changes accordingly. Zombie behaviours include:

- bite within close contact range. A zombie has to grab a survivor first before biting;
- grab within casual contact range; and,
- flock, swarm, and/or pursue survivors. For zombies, the utility of warm (living) meat is greater than that of dead meat; therefore, as soon as a survivor dies (or becomes a zombie), the zombie will pursue the nearest survivor within the zombie perception range.

The number of bites a person receives reduces the infection incubation time.

Within this zombie ED model there were a number of interventions modeled. These included:

- doctors can ‘purify’ patients that are exposed but not yet zombies. This was controlled by two parameters: the probability of a doctor shooting an exposed patient, and the probability of misdiagnosis or false positive (resulting in shooting susceptible or not shooting exposed).
- a second intervention is changing the probability of patients and HCWs being armed. Being armed is somewhat analogous to being immune, as would be the case during an influenza outbreak where the vaccinated person would not be as susceptible to infection. Carrying a shotgun, however, also allows for a person to destroy a zombie, which is without a good analogue within an influenza model.
- a third intervention called ‘wild west’ allowed anybody carrying a gun to shoot any suspected exposed survivors, including HCWs (basic anarchy).

A screenshot of the visualization of the model is shown in Fig. 20. Here various agents are illustrated ranging from patients, HCWs, infected patients, and zombies. In some cases, the patients and HCWs are carrying guns.

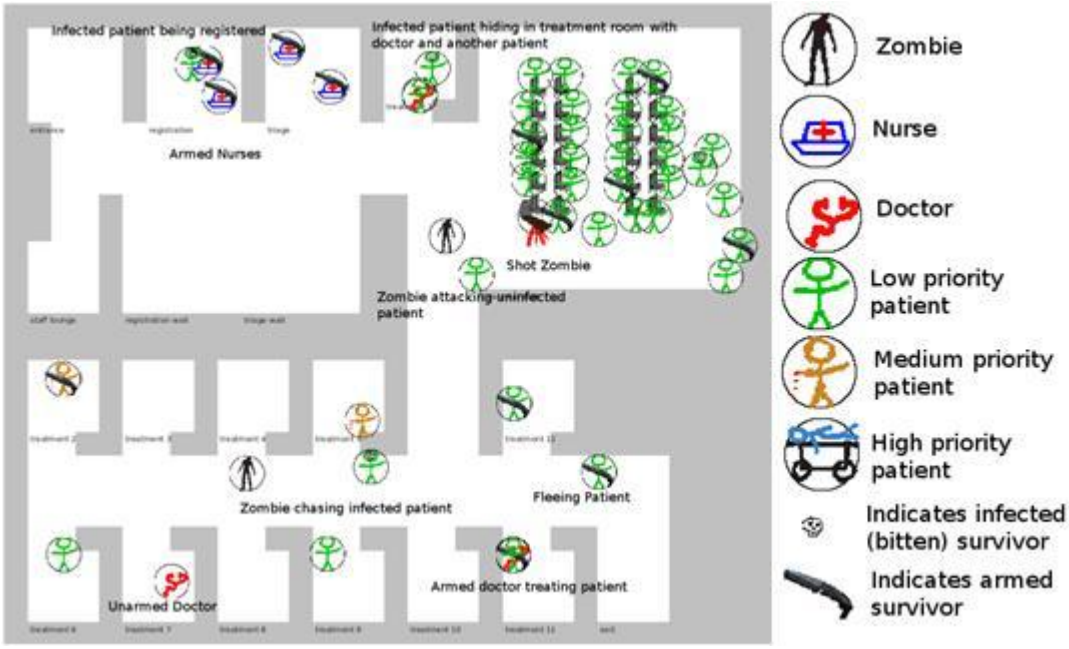


Figure 20. Screenshot of the zombie ED model

The visualization of the institutional-level model can be very informative and helps to validate the simulation. To be statistically meaningful, simulations have to be run a significant number of times. The visualizations, however, help to provide insight into rare events and outlier results (events which lie at the edge of the probability density field). These types of events are obscured in statistical summaries but are easy to discern during animation (visualization). For example, when a waiting room is quite crowded and a patient flees upon encountering a zombie, this may occasionally trigger a stampede of uninfected patients (mass fleeing). This type of rare event becomes obvious during an animation, and it may be the type of event that, although rare, has to be provisioned for by a policy maker or administrator. An advantage of using an ABM is the ability to

record and reset to initial conditions when a rare event is detected. This has obvious debugging utility but also allows one to investigate the sequence of events that precede or lead up to an anomalous state that may warrant further investigation. The ABM methodology is relatively unique in this regard. In many cases, where one is modeling a complex dynamical system, one may be interested in typical behaviour. In general, the evolution of probability density functions predicts how the system evolves as well as its stability. In the case of the ABM, the outliers (instances that would otherwise be considered statistical anomalies) may, in fact, reveal some of the more interesting dynamics of the system itself. This notion is consistent with an information theory perspective, where greater uncertainty results in greater information content.

The following series of data reveal the impacts of various policy interventions. Figure 21 illustrates the average difference in number of survivors over 8–24 hours after zombies first enter the ED. This difference is interpreted as the number of survivors at the end of the hour as compared to the beginning of the hour. As such, a value of -2 would indicate that at the end of the hour, there were two fewer survivors than at the start of the hour. Consequently, the higher the difference, the better the outcome. The family of curves is parameterized by the zombie arrival rate. This rate was extracted from an urban model (developed outside the scope this thesis. In general, the zombie arrival rates are as follows:

$$Zin_m = \frac{1000}{2^m}\left[\frac{zombies}{day}\right]$$

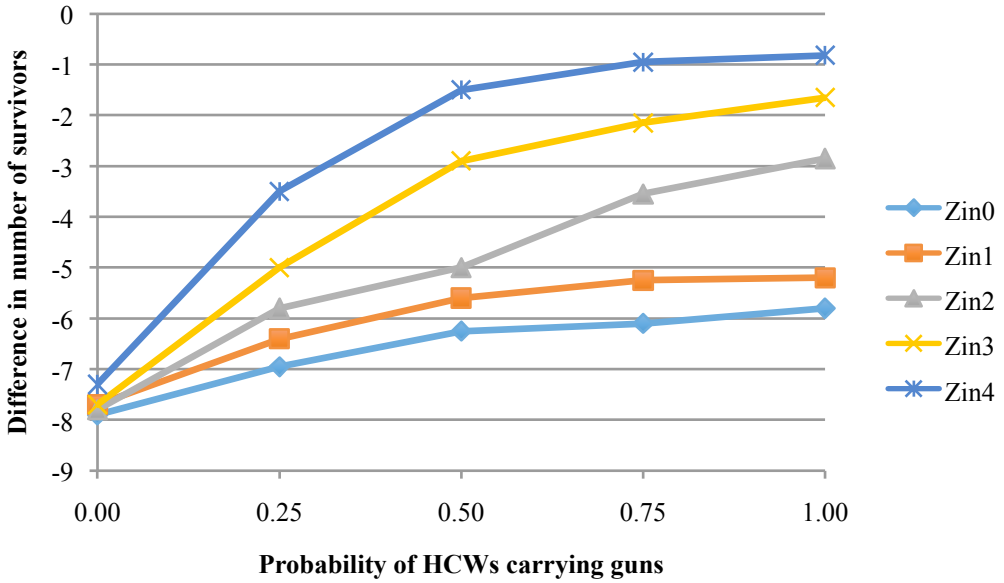


Figure 21. Survival as HCWs carrying guns varies

In this scenario, the probability of HCWs carrying a gun varied from 0 to 1. This intervention policy allows a HCW to shoot a full-fledged zombie. There was a chance of missing a zombie, but more often than not, the HCW was able to reload and successfully shoot a zombie. This scenario formed the baseline for subsequent scenarios.

Figure 22 illustrates a slightly more 'realistic' scenario, denoted as 'purification'. In this case, a ZLI-infected patient was diagnosed by an ED physician. Upon diagnosis, the ED physician could then shoot the infected patient preventing them from becoming a zombie. This scenario was in addition to all HCWs being able to shoot full-fledged zombies. Seen in Fig. 22, there is a slight reduction in acquired ZLIs as ED physicians are allowed to make a ZLI diagnosis and eradicate an infected patient.

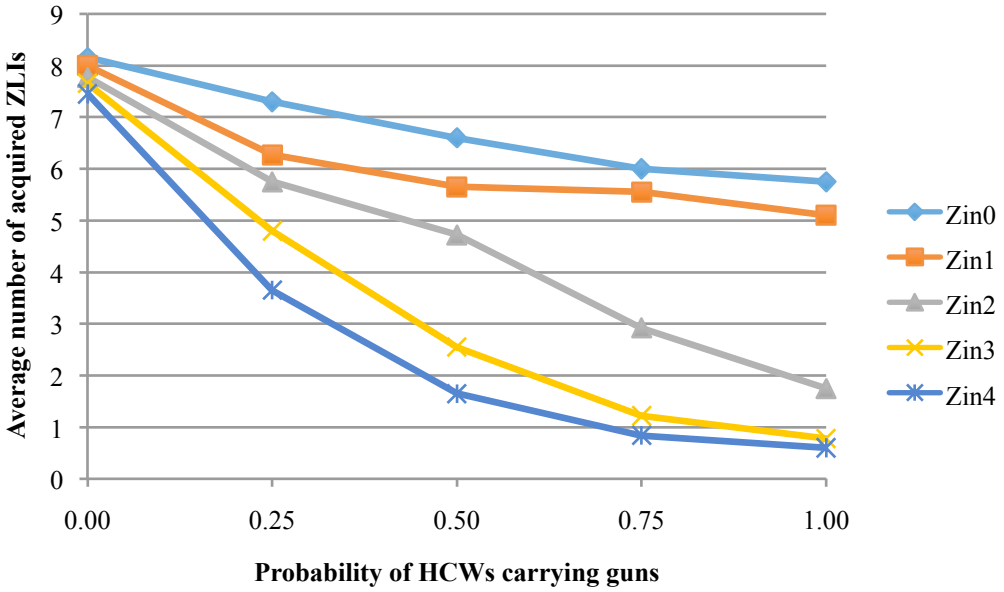


Figure 22. Acquired ZLIs as HCWs carry guns varies during 'purification'

Of course in real-life situations, there is always a consequence for irreversible actions. As a result there is a real probability of a patient being seen by a doctor and being incorrectly diagnosed as being infected with a ZLI. This false positive results in the ED physician shooting a patient erroneously. Figure 23 illustrates the average number of causalities resulting from the 'purification' policy.

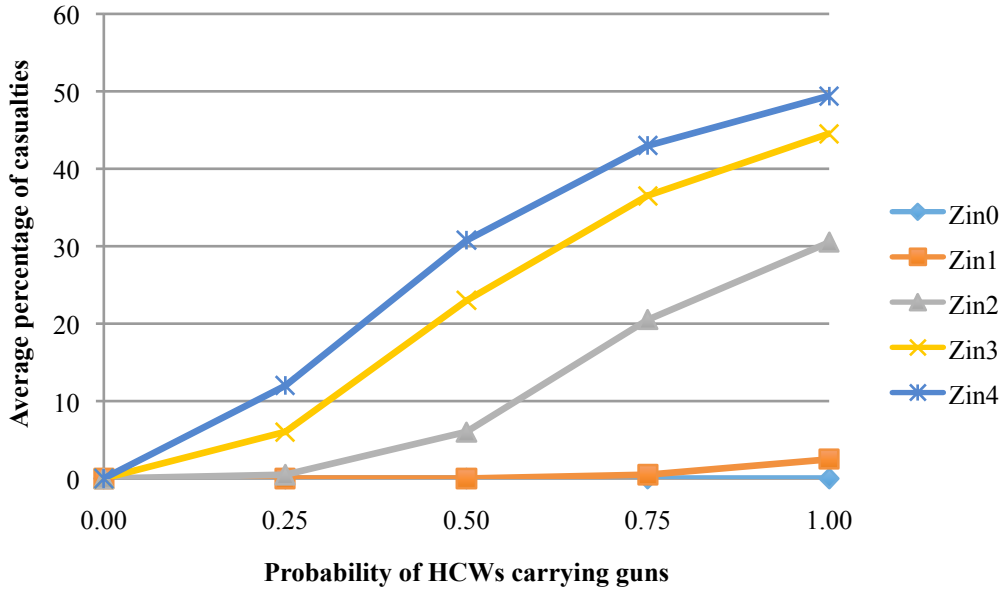


Figure 23. Causalities as HCWs carry guns varies during 'purification'

In Fig. 23, with a moderate zombie arrival rate (Zin_2), there are approximately 0.3 uninfected patients misdiagnosed, and subsequentially shot and killed, per hour. This is likely an acceptable rate within disastrous situations, where false positives are tolerated to save lives.

Figure 24 illustrates another intervention where, in addition to HCWs having guns, there is also a probability of patients carrying guns. This has been denoted as the 'wild west' intervention. This intervention is clearly very successful but relatively ineffective if the zombie-arrival rate is at its highest (Zin_0). In this case, the ED is clearly overrun with zombies.

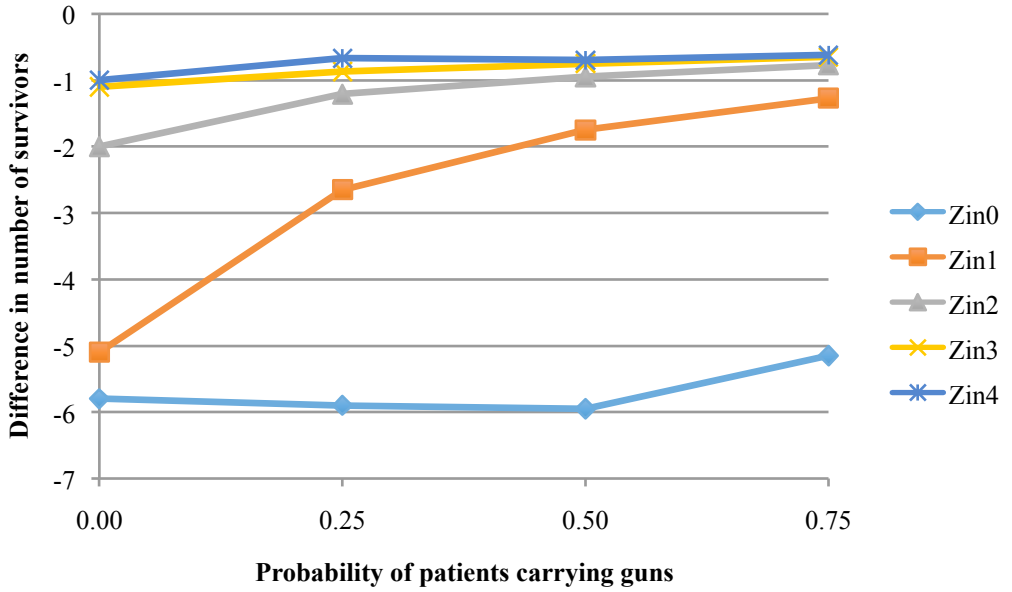


Figure 24. Survival as patients carrying guns varies during 'wild west'

Complementary to Fig. 24, there is also the concomitant increase in casualties as illustrated in Fig. 25. Once patients are armed there is also a potential to shoot an uninfected person inadvertently during the overall panic situation. In this scenario, the causality rates are higher than in the case where only HCWs carried guns. For all zombie arrival rates except Zin_0 (the highest), introducing guns to the patient population maximized the overall casualty number; exceeding that for Zin_0. This is analogous to having trained professionals (e.g., police, armed forces, compared to general untrained public) equipped with guns and supporting an official gun registry, relating to the mathematical or statistical phenomena referred to as Stein's paradox [66]. The

independent estimates of survival during a zombie outbreak support the government's estimate of the effectiveness of a gun registry.

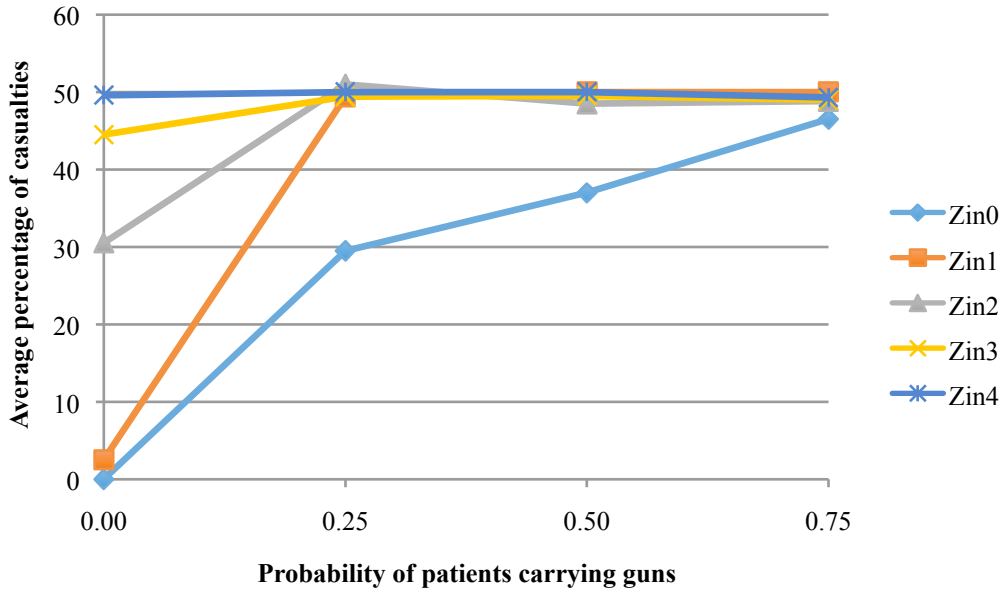


Figure 25. Causalities as patients carrying guns varies during 'wild west'

With any interventions and mitigation strategies there are also unexpected, sometimes positive side effects. Figure 26 illustrates the number of patients waiting to see an ED physician from the start of the zombies arriving through a 24-hour period.

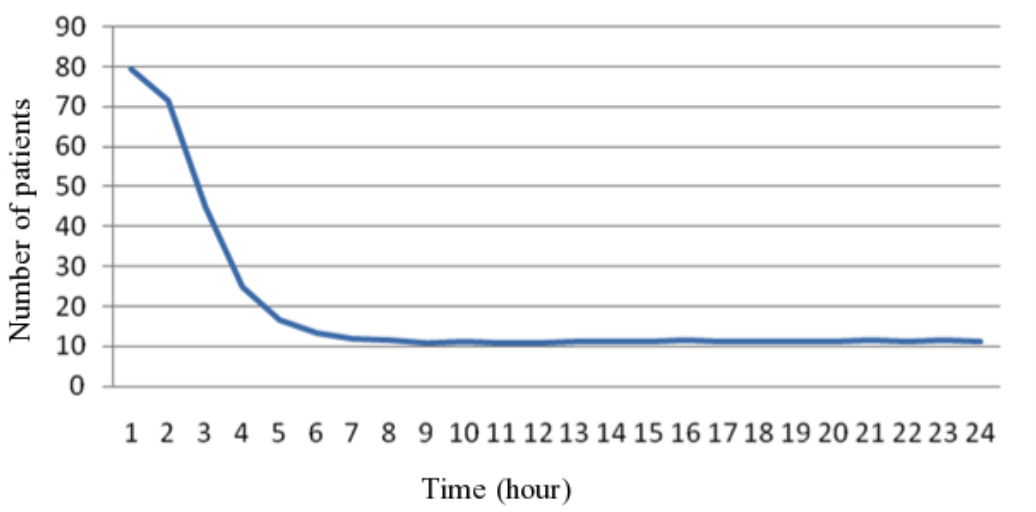


Figure 26. Patient wait time during ZLI outbreak

Among the chaos of zombies in the ED, there is a significant reduction in the number of patients and potentially a greatly reduced waiting period. The estimated waiting time was reduced from approximately 6 hours down to several minutes. This reduction in waiting time would be welcomed in any real-world ED and by the general populace.

5.4 Summary

This chapter demonstrated the ABM technique applied to a zombie outbreak in a hospital ED. One of the attributes of an ED ABM is the ability to consider ‘what if’ scenarios and to gain insight into potential policy effectiveness. The ED ABM scale was such that the agents had associated with them considerable detail in their respective behaviours. In the zombie outbreak scenario, the modeled interventions included the mitigation strategies of arming HCWs, as well as patients. Both positive and negative results were observed when implementing the various mitigation strategies.

6 Critique

6.1 Introduction

This chapter is the most significant contribution of the thesis. The previous chapters provided a literature review of ABMs within hospital environments, outlined the ABM development and simulation of contact-based infection spread in a hospital ED, as well as an ABM for modeling the spread of a nosocomial infection within an ED. That work resulted in one of the first papers published in the area [10], with the most recent being [67], further supporting the conjecture that ABM is playing an increasing role on modeling nosocomial infections.

This was followed by a chapter outlining an extended ABM for use in planning or evaluating a RFID-based RTLS for patient tracking in a hospital ED, which resulted in an InTech book chapter with an accumulated downloads to date (Jan. 3, 2016) of 4137 [68].

This was followed by a chapter about zombie-based modeling with zombies being present within an ED as an example of an ABM for disaster preparedness simulation. This model allowed for substantively greater diversity of agents and their behaviours, and experimentation with a range of non-normal agent behaviours. This collection of work demonstrated the utility of ABM within an institution.

The collective models described herein, however, are not without their shortcomings. As mentioned earlier in the thesis, custom-developed ABMs are in general of the one-off type variety. As such, although providing insight to the modeler and the environment

being modeled, they lack the attributes required for scalable and extendable software while still maintaining the advantages of software that is flexible enough to take advantages of processing advancements, such as multi-threading capabilities, or work offloading to devices such as a GPU.

The author also contributed to several projects, which showed the utility of combining real data with ABMs. In general, these projects exploited the collection of data accumulated from smartphones acting as proxies for people in order to better provide contact information to an ABM [69],[70]. Modeling that is supported by real-life data is becoming known as data-driven ABMs. Within a modern ABM framework, being able to incorporate data and perhaps lots of it would also be a current expectation.

6.2 Critical Review of the Framework

As earlier chapters mentioned, the framework reviewed here was originally inspired by a C++ framework initially written by Dr. Marek Laskowski, Ph.D, with many contributions made by the author. Some of these contributions included fixing inherent problems that caused the simulation to crash, improving simulation controls, as well as adding new agent types and behaviours. This being the case, a respectable amount of the design decisions and features that made up that initial framework were adopted into this framework as an initial starting point. The overall development plan was to start off with the same capabilities and functionality as the original framework, and slowly abstract away and create more generic and general-purpose interfaces, components, and classes.

These interfaces, components, and classes would take the place of the current, concrete, model-specific parts of the original framework.

Following this replacement, those same concrete, model-specific parts would be rewritten, using the framework as the appropriate parts were developed. These rewritten parts would then be included with the framework as an example model or even a template that could be used to create similar models of the same type or variety.

This initial starting point was also a good opportunity to consider and tackle other problems that were apparent in the original work, and incorporate solutions to those problems into the framework. These included separating out or abstracting away specific dependencies, whether from environment, development, or code, as well as tweaking any of the original design considerations that were found to not be a good fit in a more generic framework.

The following sections will go over some of the different areas that were altered and improved upon in this framework, the reasons behind making those decisions, as well as the consequences that came in doing so.

6.2.1 Environment

The development and runtime environment were primarily left as-is and were not changed. It was still a 64-bit Linux operating system running on an Intel processor architecture machine. This was primarily due to the aforementioned environment being one of the more common environments used in the lab where the framework was

developed and run. Although this was an acceptable environment to develop on, it led to a framework that, although undesired by the author, was implicitly coupled to that environment. Some of the following sections expand more on this issue, as well as have their own smaller contributions to an overall solution for this problem.

6.2.2 Platform Independence

Even though platform independence was outlined as one of the main features and end goals of the framework, there were never any substantial measures taken during development to ensure that what was being written would, in fact, run reliably in an expected fashion, run at all, or even, at the bare minimum, compile on any other environment or architecture.

This became apparent when an attempt to use the framework on an Intel processor-based machine running the Mac OSX operating system was performed. The framework did not compile successfully as expected. Even though the fix that led to a successful compilation was straightforward and trivial, the fact that the framework user had deploy the fix showed that the platform independence was not at a level where it needed to be. Later, compiling on the same environment with some of the optional features and dependencies enabled was also unsuccessful, which once again demonstrated the lack of platform independence.

Ideally, the framework was targeted to compile and run reliably on three popular operating systems: Windows, Mac OSX, and Linux. Compilation and execution on the

latter two were obtained with minimal effort. The former, however, would be more involved. Even though supporting the Windows platform was not imperative, the author felt that avoiding it would miss a large user base, as well as never fully achieve the goal of having a true platform independent framework. Even though this was a desired outcome, it was never fully achieved.

6.2.3 Dependencies

The original framework relied quite heavily upon the Qt framework (version 4). Not only was the bundled qmake build tool used to aid in compilation and building, but also many of Qt's core classes were invoked extensively from within the original framework's core classes. In doing so, depending so heavily on the Qt framework created a few potentially unwanted limitations in the eyes of the author.

The first potential limitation was that Qt, a framework primarily used for developing graphical user interfaces, was utilized in core code, which the author desired to be only dependent on the core language (C++) itself. This in no way meant that Qt could not be used at all in the framework, in fact, quite the opposite. The useful parts included from Qt were reworked back into the framework through a different mechanism (this will be expanded upon later).

The second potential limitation was that coupling the framework to Qt would implicitly restrict any other means of visualization. The reason for this restriction would be because trying to conform a different (visualization) library's interfaces and classes to adhere to

what Qt defined might end up being very difficult to do. Again, this meant that the (visual) aspects of what Qt supplied were reworked back into the framework by a more modular means.

The process of figuring out how to decouple dependencies revealed the potential for adding new dependencies with relative ease. These dependencies could be based on what was being modeled, popularity, or even extensions that would allow this framework to hook into other ABM frameworks or tools. This notion of decoupling dependencies was eventually taken to the extreme point of trying to abstract away the Standard Template Library (STL) from the framework. Not only did this result in extra work, it also led to the creation of interfaces and classes that were basically independent of ABMs and ABM frameworks, and outside of the scope of this project.

6.2.4 Build Process

As mentioned earlier, the original framework made use of qmake to aid in the build process. Since Qt was now a decoupled dependency, one of which that was used more compartmentally and less extensively, the assumption was that an alternative build tool would be needed. This assumption was based off the fact that qmake came as a bundled tool within the Qt framework. Since the Qt framework was now potentially optional, then the implication was that qmake would be optional as well. This assumption turned out to be wrong in the end, but nevertheless, CMake was chosen and used as an alternative means of building the framework due to its cross-platform design, and extensibility.

Using a new build tool resulted in an expected learning curve to become accustomed to it. Although basic functionality was picked up and used fairly quickly, some of the (slightly) more advanced features took additional time and effort. This resulted in distracting focus away from code development for the framework. Even though this was the case, it was determined to be a worthwhile exercise. Becoming familiar with CMake provided the groundwork for the framework to incorporate new dependencies, as well as target different platforms in the future.

6.2.5 C++ Standard

One unfortunate aspect about this framework was the timing at which it was being developed. A large portion of the framework code was written around the same time as the release of the new (at the time) C++11 standard. This standard included a multitude of new language features and core functionality that were not incorporated into what constituted the framework at that time. As the framework grew, some new languages features were gradually incorporated. Some of these features included using new keywords, constructor delegation, as well as range-based for loops. A majority of the framework, however, still used the previous C++03 standard.

6.2.6 Component-based Design

The original framework basically had all agent behaviours built in to each individual agent. This made it very difficult, if not impossible, to reuse similar behaviours in new or

other models. One could circumvent this through the copying and pasting of code at the potential expense of maintainability, however, this was an undesirable trade off.

This very tight coupling could also lead to difficulties when attempting to incorporate new behaviours in a pre-existing model. Any repeated manipulation or rewriting of the same or similar sections of code could introduce new bugs in sections that, up until then, functioned normally. This trade-off for time and consistency was also undesirable.

This framework was, therefore, designed to be component-based. The goal was to allow the addition and removal of lightweight components, which would each encapsulate their own, well-defined behaviour. Since each component or behaviour could vary greatly from one to another, the component interface and base class within the framework was made extremely generic.

This approach was attractive in many ways. First and foremost, it was very flexible. It allowed the modeler the freedom to add and remove components at any time. This also allowed the further extension of having conditional components be added or removed depending on other criteria (e.g., simulation state, time, or location). This ability was a desirable feature, especially in modeling more advanced systems. Not only did this approach yield easy, or at least easier, maintainability and extensibility, but also allowed for a straightforward means for conventional (unit) testing.

Another attractive quality of this approach was how it could potentially exploit a parallel environment (parallelism will be expanded upon later). Individual component behaviours could be executed concurrently rather than sequentially, thus reducing overall

execution time. This did impose certain assumptions to be made on each component, discussed later.

This design came with its own problems and limitations, as well. The trade-off of the generic component-based system was just that, it was too generic. It lacked some structure that may be necessary to ensure that any transferring of components, whether via any internal mechanisms of the framework itself, a model, or any user customizations, would not result in unresponsive agents, or an incorrectly configured model.

Since components each included their own data or state, the question arose of what would need to be done when components needed other components' data to execute, or at least aid in executing their own behaviours. This led to the ability to search for a component specified by name or type.

Searching for components was convenient, and conformed to allowing components to be swapped in and out as the simulation progressed. However, frequent searching was somewhat cumbersome and potentially slow. Each search became more inefficient as the number of components grew. Caching 'common' components could circumvent performing a costly search, however, this caching had its own drawbacks. One such drawback would be the extra precautionary measures needed to ensure any caches were properly maintained.

Another potential issue was dealing with whether one or more components contradicted one another. An example of this would be when two components decided that their agent should move in equal, but opposite directions. A decision layer would required to resolve

any potential contradictions, or certain limitations would need to be built into the framework. One such limitation could involve denying the addition of a component, based on a certain type or behavioural aspect. How to categorize components was another question to answer, and no definitive answer was ever settled upon. In the end, both approaches were not ideal and this issue was left to the responsibility of the modeler to ensure that the components used were behaviourally distinct.

Yet another issue was how to deal with components that relied on the presence of other components to function, what to do if one or more of those required components were not found, and how to handle this scenario. This was again left up to the modeler to ensure that everything that was needed would be present. These were only some of the questions that surfaced when considering components within a sequentially executed environment. Even more considerations are needed if components are to be executed in parallel.

6.2.7 Policies

To attempt to repair some of the problems with the component-based design, the concept of policies and refinements to components were introduced. Policies were developed as a more complex component-like structure, which would be active or proactive in behavioural decision-making, where as, components were refined to be more passive or reactive in functionality.

This partially solved the problem with conflicting components, however, it simply moved the problem to the policy level. The introduction of policies did aid in solving the

problem with component dependencies in the sense that each policy could be an aggregation of components that were somehow related. By having an agent subscribe to one or more policies, which would in turn add any components associated with that policy, the question of whether or not a particular component would be available (related to that policy) could be avoided.

In hindsight, the concept of policies did help with a few of the apparent problems found with components, however, most of the same deficiencies that plagued components also were applicable to policies.

6.2.8 Memory Management

The original framework suffered from a number of memory-related issues. Most of these issues were exposed and rectified by the author, who in turn gained some valuable insights on how some of these issues originally developed. It was decided that a memory subsystem would be implemented with a basic, straightforward approach that tried to follow the Resource Acquisition Is Initialization (RAII) programming idiom. This basically meant that any object or class that allocated a new resource (e.g., file, socket, graphic) was also responsible for the deallocation or cleaning up of that resource once it was no longer needed. This resulted in multiple factory classes that were responsible for the creation and, following the RAII idiom, destruction of objects.

The simplicity of this approach resulted in no action being required from the end user to manage memory. This approach, however, led to large collections of objects

accumulating and being stored for the duration of the entire simulation. These objects were only being destroyed once their producing factory was destroyed upon the completion of the simulation. In large systems with hundreds or thousands of agents or objects, each having multiple policies and/or components, this approach subjected itself to potential out-of-memory exceptions.

A more dynamic approach was introduced, which allowed the caller to flag potential objects for destruction. This mechanism did help avoid potential out-of-memory exceptions by allowing memory to be freed during the course of the simulation. This mechanism also came at the expense of being slightly more tedious or inconvenient to use. The responsibility was put on the caller, who needed to ensure the appropriate clean up function(s) were executed at the appropriate time. This approach was included as a more advanced feature, one of which could be harnessed if needed.

A final refinement that was introduced late in the development stage was the use of ‘smart pointers’, which were used in conjunction with the dynamic approach mentioned above. This was probably the most optimal approach. Full implementation required replacing pointers to dynamically allocated objects, with their wrapped ‘smart pointer’ counterparts. This modification was never fully deployed due to a drastic change of the API; it also made using the API feel slightly impure (using objects wrapping pointers instead of the pointer types themselves).

6.2.9 Decoupling and Cohesion

As time went on and the framework grew, some questions surfaced about what object/class would be responsible for doing certain functions. As more code was written, some earlier decisions about certain classes were deemed to not have been the optimal or best decision, as the purpose of the class was beginning to take on too many responsibilities. The sole purpose of the object became muddled or lost in the multiple functionalities it supplied. From a maintainability (as well as testability) standpoint, it is always better to have small, simple classes that are responsible for a single function or behaviour. As more complex functions and behaviours are needed, they can be built using the aggregation of smaller, better-defined pieces. The question arose as to whether or not 'simulation' classes (classes responsible for the simulation to run) should be independent from 'simulated' classes (classes of the things that were actually being modeled, which ideally would be anything the developer wanted to model).

6.2.10 Parallelism

The original framework was sequential in its execution. The only form of explicit parallelism exploited was running multiple, independent simulation processes at the same time. These processes were run on both the same, and separate machines in a Monte Carlo fashion. This approach yielded collecting data in a more timely manner. This approach also did not scale well if what was being modeled or simulated grew in size or complexity. This could only be achieved through a finer-grained parallel approach.

With finer-grained parallelism in mind, the framework initially had parallel build targets added to the build system to allow for different parallel technologies to be conditionally included. Some of these APIs included OpenMP, MPI, and CUDA (for GPU processing), as these were most familiar to the author at the time.

The main idea behind this was to create parallel-specific implementations of certain classes associated to each different parallel API, and then allow the end user to be able to pick and choose which implementation(s), if any, that they wanted to use based on their needs. Each build target associated with all three APIs mentioned earlier was successfully incorporated into the build system. Each build target did, however, only include a test class to ensure compilation and execution was successful opposed to any concrete, parallel-specific framework implementations.

Another consideration was to automatically use any sort of parallel implementations by default (if the target platform was equipped well enough for such processing), and allow the end user to be able to pick and choose the technologies they wanted to turn off instead. This notion was only conceptual, as it required the build process to systematically declare these types within the code, which was never established.

As with any form of fine-grained parallelism, certain considerations were needed to ensure objects behaved as expected. To make an object or class thread-safe required the use of synchronization, especially if data were being read from and written to at the same time. Another approach to thread-safety would be to make a class immutable, or read-only. All core framework classes were initially written with minimal thread-safety in

mind, due to that being an easier or more straightforward path to follow and get the bare essentials functioning. Even though the framework currently lacked in parallel-friendly classes, the groundwork was laid so it could be easily added in the future.

6.2.11 What would Java do?

The Java programming language was, and still is, where most of the development experience of the author lies. When a question about a design decision or assumption surfaced that the author was unsure about, another question was (whimsically) asked first; 'What would Java do?' (WWJD). This was perhaps the wrong question to ask, as 'What would C++ do?' seems like an obviously better fit.

Nevertheless, certain syntax and naming conventions ended up feeling more Java-like, as opposed to C++-like. Even though this is somewhat subjective and comes down to personal taste, it does potentially limit the attractiveness of the framework from the perspective of a pure C++ developer. One thing learned by the author through the use of other libraries and frameworks was that if one of those libraries or frameworks did not follow the standards or conventions of the language, it made it frustrating, or even difficult, to use in one's own development work.

6.2.12 Extraneous Code

Utility classes, functions, and other mechanisms undoubtedly come up as software is developed. These utilities are commonly included with the software that uses them for convenience purposes. Some of the problems mentioned earlier in this chapter were

abstracted out into specific interfaces that initially acted as placeholders. These placeholders were to be filled in and implemented later when time permitted. When it came time to fill in some of these gaps, some interfaces were further decomposed into more interfaces, which in turn, created more gaps to be filled.

This process continued until there were numerous small groups of interfaces, most of which were generic and somewhat unrelated to ABMs, in general. These included well-known design patterns, containers, as well as other general-purpose classes. Other interfaces and classes were implemented and included just for completeness by the author, even though they were never originally needed or used when this initial abstraction took place. This not only took focus away from the framework itself, it also introduced new questions and potential problems not necessarily within the original scope of the framework.

6.2.13 Visualization

As previously mentioned, the original framework made heavy use of Qt. This provided an easy and logical step to implement any visualization using Qt as well. This also meant that the original framework would always run and include a visual element, which not only increased overall execution time, but also computationally wasteful depending where the simulation ran. An example, such as a computing cluster or grid where no one would be watching the simulation executes.

Having a visual aspect in the framework is still desirable. Not only can visualization provide a quick way to verify that things ‘appear’ to be working properly or as expected, but it can also expose strange looking behaviour or potential bugs much more quickly if those problems translate to an odd or strange looking action. A visual aspect, component, or tool can also give the software a more polished look and feel, which can entice more users to embrace the framework. Since visualization was still an import aspect to retain, the framework allowed for it through the inclusion of a visualization component. One such component could be tied to a Qt implementation, where another component could be tied to a different visualization API or library (e.g., Unity).

A lot of discussion and thought took place about the different kinds or types of visualization that could be utilized, and how to go about doing so in conjunction with the other new design features that were added to the framework. Some of these considerations included direct access, a client/server architecture, and action recordings. The direct access method involved a component that simply wrapped another visualization object. The client/server method would involve a ‘client’ component sending information to a ‘server’ via some connection, such as a socket. The recording method would involve writing events to a log or file, which could then be ‘played back’ at a later time. These methods (other than the direct access method) were conceptual, and never evolved beyond that point.

6.2.14 Model Creation

In the original framework, the layout and design of the model was initially always the same. It was created programmatically when the simulation was first started, with dimension and location values all hardcoded in place. As mentioned in an earlier chapter, one of the contributions added by the author was to allow different layouts to be constructed through a 'builder' option of the original framework. This builder was very simple in its construction, and it allowed a user to specify a few very basic constructs (e.g., walls, rooms, other locations). This created layout could then be exported to a file and loaded in subsequent runs.

Allowing the construction of a model from a pre-made configuration file or layout was an important feature to preserve in the framework. The only drawback to preserving this feature was that the builder needed to be kept up-to-date as new objects and types that could be modeled were created. Contrary to this, the builder could also be updated or rewritten to incorporate the ability to inspect the runtime types and attributes of objects and classes that could be modeled. As well, the manipulation of those attributes in order to create the appropriate objects and types with those specific values would be required. This approach would obviously require a runtime type inference system, which was incorporated into the framework as an experimental feature and never fully utilized anywhere. Unfortunately, neither approach was taken, so the original builder was still tied to the simple constructs that it was initially created with.

6.2.15 Testability

One important and often overlooked aspect of any development work is testing. Testing not only provides early detection of potential problems and bugs, but it also gives insights into how well an API is laid out or how straightforward it is to use. Much of the contributions added by the author to the original framework were to fix problems that were present in part due to a lack of any actual tests that may have caught those same issues earlier on in its development.

With that in mind, a few simple test classes were included in the framework to ensure that core classes would behave as expected. Even though this was a good starting point, the number of tests decreased as more software was developed, due to procrastination of the author. Much of the framework fell short on the testing front, resulting in the overall reliability of the framework being questionable at the least.

6.2.16 Documentation

One of the most frustrating things in the eyes of the author, and possibly other developers, was finding a tool or library that claims to help in solving a specific problem or perform a specific task, yet then does not provide any documentation or examples about how to use that resource to achieve the end goal. This framework was unfortunately no exception. Again, most of the time and effort went into writing code, and any documentation was to be filled in at a later date. Any user who was unfamiliar

with the source code would have a difficult time in trying to use the framework as it currently stands, due to lack of documentation.

6.3 Recommendations

As mentioned earlier, there were many design decisions and assumptions made during the development of the framework, some of which were not the best choices in the long run. There were many features and goals set out early on in the framework as well, that never materialized into anything substantial. This section covers some recommendations about what would be done differently to overall improve the framework.

6.3.1 Overall

This framework was a very ambitious project, which contained a lot of ideas and new features that were desired for the end result. This was also a downfall, as time and effort was divided between many various features, without enough focus on any individual one to see it through to completion. This resulted in a portion of the foundation work being done, however, much of its implementation was left incomplete.

6.3.2 Environment and Platform Independence

The framework was developed and run on only one environment. When it came time to compile and run the framework on another environment, it failed. Although there were numerous other steps that could have been done to achieve an environment-independent framework, as a recommendation, it would be beneficial to compile, run, and test

multiple environments as early in the development process and as frequently as possible in order to avoid similar issues in the future.

6.3.3 Build Process

The build process was improved overall by allowing for the addition of both new dependencies and target platforms. This aspect of the framework, although not directly related to the framework code itself, was one of the things that the author ended up being pleased with overall. The only recommendation for this section would be to potentially explore more of the advanced features that the current build system provides.

6.3.4 C++ Standard

The framework unfortunately missed out on most of the new language features included in the C++11 standard. One recommendation would be to revisit any code written before the standard was released and incorporate any new features consistently throughout those classes, as opposed to only classes written at a later time. Examples of this include implementing a move constructor in classes where applicable and incorporating threading functionality to achieve finer-grained parallelism.

6.3.5 Component-based Design

There were many inherent problems with the component-based design. Some of those problems included the component system being overly generic, as well as how to pass data between components when needed. One recommendation would be to have all

components read and write their data to another object (e.g., attributes map). This would allow components to execute in parallel, with any thread-safety being left as a responsibility of that other object. This would also allow the data associated with specific components (e.g., position coordinate) to be looked up and referred to using a component-independent type or identifier, such as the string ‘position’.

6.3.6 Memory Management

There was an initial memory subsystem put in place within the framework that evolved from a no-action, simple approach to a more refined methodology allowing dynamic memory management. A final refinement was proposed to be the most flexible but was never completely carried out. Most of the code is already there and ready to use, so making that transition would be fairly straightforward. Even though resulting in a more ‘impure’ API, other techniques could be used in conjunction with this to ease that feeling (such as type-defining or ‘typedef’-ing). This would give the final API a more transparent look and feel when in use.

A third-party memory management library or framework could be used instead of using the one written by the author. Ideally, this library could be swapped into place without much work to incorporate it. The only reason to complete this would be the assumption that a dedicated library would do a much better job than something created as an accompanying feature of a different type of framework. The accompanying research and testing would also need to be done to further support swapping out implementations.

6.3.7 Parallelism

The groundwork for fine-grained parallelism was added to the framework, however, there was a lack of parallel-specific implementations for the end user to utilize. An obvious recommendation would be to provide those parallel-specific implementations. Even though specific parallel APIs were mentioned earlier (i.e., MPI, OpenMP, CUDA) and would still desired for inclusion, targeting the new threading capabilities in the C++11 (and later) standard would be more beneficial. One benefit being that implementations depending on the threading capabilities included in the standard would appeal to a majority of users when compared to implementations that rely on potentially optional dependencies.

6.3.8 What would Java do?

A majority of the author's development is in the Java programming language. This resulted in the framework conforming to some of Java's conventions and standards (e.g., class and function names) opposed to those of C++. Although this has no effect on the overall functionality of how the framework operates, developers (especially pure C++ developers) may find it frustrating to use. It may be beneficial to rewrite parts of the framework to completely adhere to C++ conventions, but this recommendation would only be done if there were enough pressure from the user-community to do so.

6.3.9 Extraneous Code

Utility classes and interfaces were originally written as placeholders to be filled in at a later time. Some of these were further decomposed into more classes and interfaces, which needed to be eventually implemented as well. Some interfaces were added in for completeness, even though they were outside of the scope that led to the original abstraction to take place. As a recommendation, it would be beneficial to try to keep the framework as simple and straightforward as possible. Abstracting certain problems out by the use of interfaces may be unavoidable, however, there may exist other libraries or frameworks that can be utilized for a quick implementation if needed. Another recommendation would be to spend more time trying find a library that provides the needed functionality, even if that functionality is temporary. This will allow the framework development to keep progressing without getting side tracked on unrelated problems or issues.

6.3.10 Visualization

Visualization was an important feature to retain in the framework. There was a lot of time and thought put in to different methods for implementation, but in the end, most if not all of these ideas remained conceptual. As a recommendation, any visualization would now be pulled out into a specific standalone tool. The mechanism to incorporate this feature into the model would still be component-based, where that component would simply write or log an event to a file in a format that the external visualization tool could

interpret. The tool would then be able to load that file at a later time, and basically ‘play back’ all the events that took place.

6.3.11 Model Creation

Model creation from an external configuration file or layout was possible within the framework, however, the problem was how to generate the file itself. As a recommendation, a separate tool would be utilized for creating models. The tool would incorporate all the classes that could be modeled, as well as the functionality to manipulate any data contained within. This tool would rely on a runtime type inference system, which would also need to be completed along with the tool.

6.3.12 Testability

Initial testing started off strong, however, never kept up with the overall pace of the framework development. This was in part due to each test class being a custom class, as well as some of the functionality and behaviour was not entirely stable; any tests related to this was simply left out until the functionality or behaviour was stable. As a recommendation, one of the many testing libraries would be used to supply adequate tests for the framework. Not only would the testing framework allow tests to be written more easily, it would allow testing to keep pace with framework development.

6.3.13 Documentation

Documentation was non-existent throughout the framework. This resulted in a framework that, without prior knowledge or a lot of source code reading, would be difficult for a new user to pick up and use. As a recommendation, documentation would be made a higher priority task than what it was. All classes and functions would have appropriate definitions, clear and concise descriptions of arguments and behaviour, as well as any side effects or caveats that resulted in their use or execution. As a recommendation, documentation would be added at the time the class or function was declared, when the initial purpose would be most apparent.

6.4 Summary

This chapter critiqued and offered solutions to some of the initial problems in design as well as structure, which were implicit within the original framework. Each problem was expanded upon, accompanied with an attempt of how each problem was rectified within this framework. This was concluded with how each problem could be handled if they were experienced today.

7 References

[1] B. Demianyk, and D. Sandison, *Agent-Based Model Synthesized Contact Graphs for Use in Disease Control,* Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, 2010.

[2] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," *Proceedings of the National Academy of Sciences,* vol. 99 (Suppl 3), pp. 7280-7287, 2002. [Online]. Available: http://www.pnas.org/content/99/suppl.3/7280.full#xref-ref-3-1 [Dec. 22, 2015].

[3] N. Hupert, W. Xiong, and A. Mushlin, "The virtue of virtuality: the promise of agent-based epidemic modeling," *Translational Research,* vol. 151, no. 6, pp. 273-274, 2008.

[4] A. Borshchev, and A. Filippov, "From system dynamics and discrete event to practical agent based modeling: reasons, techniques, tools," in *Proceedings of the 22nd International Conference of the System Dynamics Society,* Oxford, England, 2004, p. 45 [Online]. Available: http://www.systemdynamics.org/conferences/2004/SDS_2004/PAPERS/381BORSH.pdf [Dec. 22, 2015].

[5] N. Mustafee, K. Katsaliaki, and S.J.E. Taylor, "Profiling literature in healthcare simulation." *Simulation,* vol. 86, no. 8-9, pp. 543-558, 2010.

[6] D.A. Luke, and K.A. Stamatakis, "Systems science methods in public health: dynamics, networks, and agents," *Annual Review of Public Health,* vol. 33, p. 357, 2012.

[7] P. Escudero-Marin, and M. Pidd, "Using ABMS to simulate emergency departments," in *Proceedings of the 2011 Winter Simulation Conference,* Phoenix, AZ, 2011, pp. 1239-1250.

[8] 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: Springer-Verlag, 2013, pp. 45-74.

[9] S.S. Jones, and R.S. Evans, "An agent based simulation tool for scheduling emergency department physicians," in *AMIA Annual Symposium Proceedings*, Washington, DC, 2008, pp. 338-342, [Online]. Available: http://ncbi.nlm.nih.gov/pmc/articles/PMC2656074/ [Dec. 23, 2015].

[10] M. Laskowski, B.C.P. Demianyk, J. Witt, S.N. Mukhi, M.R. Friesen, and R.D. McLeod. "Agent-based modeling of the spread of influenza-like illness in an emergency department: a simulation study," *IEEE Transactions on Information Technology in Biomedicine,* vol. 15, no. 6, pp. 877-889, 2011.

[11] J.R. Hotchkiss, P. Holley, and P.S. Crooke, "Analyzing pathogen transmission in the dialysis unit: time for a (schedule) change?" *Clinical Journal of the American Society of Nephrology,* vol. 2, no. 6, pp. 1176-1185, 2007.

[12] J.R. Hotchkiss, D.G. Strike, and P.S. Crooke, "Pathogen transmission and clinic scheduling," *Emerging Infectious Diseases,* vol. 12, no. 1, p. 159, 2006.

[13] W. Rand, and R.T. Rust, "Agent-based modeling in marketing: guidelines for rigor," *International Journal of Research in Marketing*, vol. 28, no. 3, pp. 181-193, 2011.

[14] R. Sibbel, and C. Urban, "Agent-based modeling and simulation for hospital management," in *Cooperative Agents: Applications in the Social Sciences.* N.J. Saam and B. Schmidt, Eds. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2001, pp. 183–199.

[15] A. Daknou, S. Hammadi, H. Zgaya, and H. Hubert, "Agent-based optimization and management of healthcare processes at the emergency department," *International Journal of Mathematics and Computers Simulation,* vol. 2, no. 3, pp. 285-294, 2008.

[16] A.K. Kanagarajah, P.A. Lindsay, A.M. Miller, and D.W. Parker, "An exploration into the uses of agent-based modeling to improve quality of health care," in *Proceedings of the Sixth International Conference on Complex Systems,* Boston, MA, 2006, pp. 471-478.

[17] D. Blachowicz, J.H. Christiansen, A. Ranginani, K.L. Simunich, "How to determine future EHR ROI: agent-based modeling and simulation offers a new alternative to traditional techniques," *Journal Healthcare Information Management,* vol. 22. No. 1, pp. 39-45, 2008. [Online]. Available: http://www.himss.org/content/files/jhim/22-1/11.pdf [Dec. 23, 2015].

[18] T.J. Coats, and S. Michalis, "Mathematical modelling of patient flow through an accident and emergency department," *Emergency Medicine Journal,* vol. 18.3, pp. 190-192, 2001.

[19] T. Ruohonen, P. Neittaanmaki, and J. Teittinen, "Simulation model for improving the operation of the emergency department of special health care," in *Proceedings of the 2006 Winter Simulation Conference,* Monterey, CA, 2006, pp. 453-458.

[20] C.W. Spry, and M.A. Lawley, "Evaluating hospital pharmacy staffing and work scheduling using simulation," in *Proceedings of the 2005 Winter Simulation Conference,* Orlando, FL, 2005, pp. 2256-2263.

[21] V. Mysore, G. Narzisi, L. Nelson, D. Rekow, M.Triola, A. Shapiro, and B. Mishra, "Agent modeling of a Sarin attack in Manhattan," in *Proceedings of the First International Workshop on Agent Technology for Disaster Management, ATDM, held in conjunction with the Fifth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, Future University, Hakodate, Hokkaido, 2006, pp. 108-115.

[22] M. Taboada, E. Cabrera, F. Epelde, M.L. Iglesias, and E. Luque, "Using an agent-based simulation for predicting the effects of patients derivation policies in emergency departments," *Procedia Computer Science*, vol. 18, pp. 641-650, 2013.

[23] E. Cabrera Flores, "Agent-based simulation to optimise emergency departments," M.S. Thesis, Department of Computer Architecture and Operating Systems, Universitat Autonomade Barcelona, Barcelona, 2010.

[24] E. Cabrera, E. Luque, M. Taboada, F. Epelde, and M.L. Iglesias, "ABMS optimization for emergency departments," in *Proceedings of the 2012 Winter Simulation Conference,* Berlin, 2012, pp. 1003-1014.

[25] M. Taboada, E. Cabrera, M.L. Iglesias, F. Epelde, and E. Luque, "An agent-based decision support system for hospitals emergency departments," *Procedia Computer Science,* vol. 4, pp. 1870-1879, 2011.

[26] M.H. Rahmat, M. Annamalai, S.A. Halim, and R. Ahmad, "Agent-based modelling and simulation of emergency department re-triage," in *Business Engineering and Industrial Applications Colloquium (BEIAC), 2013 IEEE,* Langkawi, Malaysia, 2013, pp. 219-224.

[27] M.E. Lim, A. Worster, R. Goeree, and J.-É. Tarride, "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,* vol. 13, no. 1, p. 59, 2013.

[28] J.R. Hotchkiss, D.G. Strike, D.A. Simonson, A.F. Broccard, and P.S. Crooke, "An agent-based and spatially explicit model of pathogen dissemination in the intensive care unit," *Critical Care Medicine,* vol. 33, no. 1, pp. 168-176, 2005.

[29] C. van den Dool, M.J.M. Bonten, E. Hak, J.C.M. Heijne, and J. Wallinga, "The effects of influenza vaccination of health care workers in nursing homes: Insights from a mathematical model," *PLoS Medicine*, vol. 5, no. 10, e200, pp. 1453-1460, 2008, [Online]. Available: doi:10.1371/journal.pmed.0050200 [December 23, 2015].

[30] B. Bean, B.M. Moore, B. Sterner, L.R. Peterson, D.N. Gerding, and H.H. Balfour Jr., "Survival of influenza viruses on environmental surfaces," *The Journal of Infectious Diseases,* vol. 146, no. 1, pp. 47-51, 1982.

[31] Y. Thomas, G. Vogel, W. Wunderli, P. Suter, M. Witschi, D. Koch, C. Tapparel, and L. Kaiser, "Survival of influenza virus on banknotes," *Applied and Environmental Microbiology,* vol. 74, no. 10, pp. 3002-3007, 2008.

[32] Centers for Disease Control and Prevention. 2012, July 2. *"SARS Fact Sheet,"* [Online]. Available: http://www.cdc.gov/sars/about/fs-SARS.pdf [Dec. 23, 2015].

[33] Ontario Ministry of Health and Long-Term Care. 2014, July. *"Best practices for surveillance of health care-associated infections in patient and resident populations,"* Provincial Infectious Diseases Advisory Committee, [Online]. Available: http://www.publichealthontario.ca/en/eRepository/Surveillance_3-3_ENGLISH_2011-10-28%20FINAL.pdf [Dec. 23, 2015].

[34] R.D. McLeod, M. Laskowski, M.R. Friesen, and B. Podaima, *"Agent based models for nosocomial infections: modeling of hospital-acquired infections within a hospital,"* Report for the Public Health Agency of Canada, 2009, available from authors.

[35] B.S. Ong, M. Chen, V. Lee, and J.C. Tay, "An individual-based model of influenza in nosocomial environments," in *Proceedings of the 8th International Conference of Computational Science,* Krakow, Poland, 2008, Part I, vol. 5101, pp. 590-599.

[36] Y. Meng, R. Davies, K. Hardy, and P. Hawkey, "An application of agent-based simulation to the management of hospital-acquired infection," *Journal of Simulation*, vol. 4, no. 1, pp. 60-67, 2010.

[37] E. D'Agata, P. Magal, D. Olivier, S. Ruan, and G.F. Webb, "Modeling antibiotic resistance in hospitals: the impact of minimizing treatment duration," *Journal of Theoretical Biology,* vol. 249, no. 3, pp. 487-499, 2007.

[38] J. Ferrer, M. Salmon, and L. Temime, "Nosolink: an agent-based approach to link patient flows and staff organization with the circulation of nosocomial pathogens in an intensive care unit," *Procedia Computer Science*, vol. 18, pp. 1485-1494, 2013.

[39] L. Milazzo, J.L. Bown, A. Eberst, G. Phillips, and J.W. Crawford, "Modelling of healthcare associated infections: a study on the dynamics of pathogen transmission by using an individual-based approach," *Computer Methods and Programs in Biomedicine,* vol. 104, no. 2, pp. 260-265, 2011.

[40] L. Caudill, "A hybrid agent-based and differential equations model for simulating antibiotic resistance in a hospital ward," in *Proceedings of the 2013 Winter Simulation Conference,* Washington, DC, 2013, pp. 1419-1430.

[41] D.W. Lowery-North, V.S. Hertzberg, L. Elon, G. Cotsonis, S.A. Hilton, C.F. Vaughns II, E. Hill, A. Shrestha, A. Jo, and N. Adams, "Measuring social contacts in the emergency department," *PloS One*, vol. 8, no. 8, p. e70854, 2013.

[42] M.R. Poynton, V. Shah, R. BeLue, B. Mazzotta, H. Beil, and S. Habibullah, "Computer terminal placement and workflow in an emergency department: an agent-based model," in *Proceedings of the Complex Systems Summer School Winter Simulation Conference,* Washington, DC, 2007, pp. 1-3.

[43] M. Laskowski, and S. Mukhi, "Agent-based simulation of emergency departments with patient diversion," in *Electronic Healthcare,* D. Weerasinghe, Ed. Berlin Heidelberg: Springer, 2009, pp. 25-37.

[44] M. Laskowski, "A prototype agent based model and machine learning hybrid system for healthcare decision support," *International Journal of E-Health and Medical Communications*, vol. 2, no. 4, pp. 67-90, 2011.

[45] A. Al-Refaie, R.H. Fouad, M-H. Li, and M. Shurrab, "Applying simulation and DEA to improve performance of emergency department in a Jordanian hospital," *Simulation Modelling Practice and Theory*, vol. 41, pp. 59-72, 2014.

[46] A. Anagnostou, J. Eatock, and S.J.E Taylor, "Nosopolis: towards a hybrid agent-based discrete event simulation tool for emergency medical services improvement," in *Proceedings of the 2012 Winter Simulation Conference,* Berlin, 2012, p. 365.

[47] M. Laskowski, B. Demianyk, M.R. Friesen, and R.D. McLeod, "Uncertainties inherent in RFID tracking systems in an emergency department," in *IEEE Workshop on Health Care Management (WHCM),* Venice, 2010, pp. 1-6.

[48] A. Vespignani, "Modelling dynamical processes in complex socio-technical systems," *Nature Physics*, vol. 8, no. 1, pp. 32-39, 2012.

[49] M. Laskowski, R.D. McLeod, M.R. Friesen, B.W. Podaima, and A.S. Alfa, "Models of emergency departments for reducing patient waiting times," *PloS One*, vol. 4, no. 7, p. e6127, 2009.

[50] S. Wolfram, *A New Kind of Science*, Champaign, IL: Wolfram Media, Inc., 2002.

[51] K.P. White, Jr., "A survey of data resources for simulating patient flows in healthcare delivery systems," *Proceedings of the 2005 Winter Simulation Conference,* Orlando, FL, 2005, pp. 926-935.

[52] L. Patvivatsiri, "A simulation model for bioterrorism preparedness in an emergency room," *Proceedings of the 2006 Winter Simulation Conference,* Monterey, CA, 2006, pp. 501-508, 2006.

[53] Winnipeg Regional Health Authority, "*Emergency Department Information System,*" Raw data January-June, Winnipeg, MB, WRHA, 2009.

[54] H.A. Piwowar, M.J. Becich, H. Bilofsky, and R.S. Crowley, "Towards a data sharing culture: recommendations for leadership from academic health centers," *PLoS*

Medicine, vol. 5, no. 9, p. e183, 2008, [Online]. Available: doi:10.1371/journal.pmed.0050183 [December 27, 2015].

[55] B.S. Cooper, G.F. Medley, and G.M. Scott, "Preliminary analysis of the transmission dynamics of nosocomial infections: stochastic and management effects," *Journal of Hospital Infection*, vol. 43, pp. 131-147, 1999.

[56] I. Pelupessy, M.J.M. Bonten, and O. Diekmann, "How to assess the relative importance of different colonization routes of pathogens within hospital settings," *Proceedings of the National Academy of Sciences*, vol. 99, no. 8, pp. 5601-5605, 2002.

[57] W. Banzhaf, P. Nordin, R. Keller, and F. Francone, *Genetic programming - an introduction: on the automatic evolution of computer programs and its applications,* San Francisco, CA: Morgan Kaufmann Publishers, Inc., 1998.

[58] J.R. Koza, M.A. Keene, M.J. Streeter, W. Mydlowec, J. Yu, and G. Lanza, *Genetic programming IV: routine human competitive intelligence*. Norwell, MA: Kluwer Academic Publishers, 2003.

[59] S.W. Wang, W.H. Chen, C.S. Ong, L. Liu, and Y.W. Chuang, "RFID application in hospitals: a case study on a demonstration RFID project in a Taiwan hospital," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, vol. 8, pp. 1-10, 2006.

[60] P. Kanyuk, and J. Young, *RFID in Healthcare: Novelty or Mass Market?* Datamonitor, New York, NY, Report Ref. Code: BFTC1095, 2004.

[61] A. Solanas, and J. Castellà-Roca, "RFID technology for the health care sector," *Recent Patents on Electrical Engineering*, vol. 1, no. 1, pp. 22-31, 2008.

[62] A.M. Wicks, J.K. Visich, and S. Li, "Radio frequency identification applications in healthcare," *International Journal of Healthcare Technology and Management*, vol. 7, no. 6, pp. 522–540, 2006.

[63] K. Finkenzeller, *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification,* 2nd edition, New York, NY: John Wiley & Sons, Inc., 2003.

[64] D. Sanders, S. Mukhi, M. Laskowski, M. Khan, B.W. Podaima, and R.D. McLeod, "A network-enabled platform for reducing hospital emergency room waiting times using an RFID proximity location system," in *19th International Conference on Systems Engineering,* Las Vegas, NV. 2008, pp. 538-543.

[65] A. Bhatia, B. Mehta, and R. Gupta. (2007, Aug.). "Different localization techniques for real time location sensing using passive RFID," [Online]. CiteseerX, The Pennsylvania State University, State College, PA, 2007. Available:

http://citeseerx.ist.psu.edu/viewdoc/similar?doi=10.1.1.452.5773&type=cc [Dec. 27, 2015].

[66] "Stein's example", in *Wikipedia, the Free Encyclopedia* [Online]. May 14, 2005. Available: http://en.wikipedia.org/wiki/Stein's_example [Dec. 27, 2015].

[67] C. Jaramillo, D. Rexachs, E. Luque, F. Epelde, and M. Taboada, "Modeling the contact propagation of nosocomial infection in hospital emergency departments," in *SIMUL 2014: The Sixth International Conference on Advances in System Simulation,* Nice, France, 2014, pp. 84-89.

[68] M. Laskowski, B.C.P. Demianyk, G. Naigeboren, B.W. Podaima, M.R. Friesen, and R.D. McLeod. (2010). "RFID Modeling in Healthcare," in *Sustainable Radio Frequency Identification Solutions* [Online], C. Turcu, Ed. InTech. Available: http://www.intechopen.com/books/sustainable-radio-frequency-identification-solutions/rfid-modeling-in-healthcare [Dec. 27, 2015].

[69] J. Benavides, B.C. Demianyk, S.N. Mukhi, M. Laskowski, M. Friesen, and R.D. McLeod, "Smartphone technologies for social network data generation and infectious disease modeling," *Journal of Medical and Biological Engineering*, vol. 32, no. 4, pp. 235-244, 2012.

[70] J. Benavides, B. Demianyk, R.D. McLeod, M.R. Friesen, M. Laskowski, K. Ferens, and S.N. Mukhi, "3G smartphone technologies for generating personal social network contact distributions and graphs," in *First IEEE International Conference on Healthcare Informatics, Imaging and Systems Biology (HISB),* San Jose, CA, 2011, pp. 182-189.