Methicillin resistant *Staphylococcus aureus* transmission reduction using Agent-Based Modeling and Simulation

Sean Barnes
PhD Student, Applied Mathematics and Scientific Computation
Department of Mathematics
University of Maryland, College Park, MD
sbarnes@math.umd.edu

Dr. Bruce Golden
Professor, Robert H. Smith School of Business
University of Maryland, College Park, MD
bgolden@rhsmith.umd.edu

Abstract

Methicillin resistant *Staphylococcus aureus* (MRSA) is a significant problem arising in healthcare, most commonly in large, tertiary-care hospitals, and its spread among patients causes many downstream effects, such as longer lengths of stay for patients, higher costs for hospitals and insurance companies, and in a significant number of cases, fatalities. An agent-based simulation model is developed to investigate the dynamics of MRSA transmission in the University of Maryland Medical Center (UMMC). The simulation model is used to examine the effectiveness of infection control procedures to reduce the spread of infection within the medical center. Specifically, simulation experiments are performed to examine the efficacy of hand hygiene compliance, patient screening, decolonization, patient isolation, patient cohorting, and nurse cohorting on the incidence of MRSA transmission and other relevant metrics. Preliminary testing has produced results comparable to those presented in the literature, specifically those relating to the effects of hand hygiene compliance. Specifically, a health care worker compliance of approximately 30% appears to be a critical value, below which a significant MRSA outbreak occurs and above which the transmission appears to be contained. In addition, hand hygiene compliance displays the law of diminishing returns, as further reduction in transmission requires significant increases in compliance.

Introduction

In large hospitals, there are a large number of patients as well as health care workers (HCWs) that come into contact with each other frequently throughout the course of a day. If one of those patients or HCWs becomes colonized with methicillin resistant *Staphylococcus aureus* (MRSA), the bacteria could spread by way of HCWs within the hospital. As a result, many patients fall victim to hospital-acquired, or nosocomial infection. It is estimated by the Committee to Reduce Infection Deaths (RID) that infections acquired in hospitals lead to over 100,000 deaths per year and an additional $30.5B in hospital costs [1]. More specifically, close to 300,000 (out of 2 million) infection cases involved MRSA, with close to 20,000 of those cases resulting in fatalities.

Many experts agree that hospital-acquired, or nosocomial, infections (HAIs) are almost entirely preventable [2], given a committed and capable healthcare institution. However, studies have shown that such measures have proven difficult to implement and enforce, due to both HCW non-compliance and cost considerations. These infection control policies consist of a number of measures aimed at reducing the incidence of MRSA transmission. The first measure aims at improving hand hygiene compliance of health care workers. The next policy involves screening patients for MRSA, at admission and with some frequency during their stay. This policy allows for the detection of colonized patients so that further measures can be taken to prevent transmission to other patients in the hospital.

Among these additional measures are patient isolation, decolonization, patient cohorting, and nurse cohorting. Patient isolation involves moving a colonized or infected patient to a single room, so that they are not as likely to colonize other patients. The decolonization process involves a regimen aimed at reducing or removing the presence of bacteria on the skin of patient, which is done typically through the use of antibiotics and alcohol-based bathing. Patient cohorting is an alternative to patient isolation, to be used when single patient rooms are not available. In this case, colonized patients are placed in rooms with other colonized patients so that they are not as likely to introduce the bacteria to susceptible patients. Nurse cohorting is the practice of assigning patients to nurse-specific groups, thereby decreasing the connectivity of HCWs and patients throughout the hospital, which should decrease the likelihood of transmission.

This study seeks to identify the most effective infection control measure or measures that could reduce the incidence of MRSA transmission without becoming cost prohibitive. To accomplish this goal, an agent-based simulation package is designed and developed to model MRSA transmission dynamics and investigate the impact of infection control measures (ICMs) in the University of Maryland Medical Center (UMMC).

Methodology

Historically, this problem has been approached using a number of survey and data collection techniques that evaluate a combination of one or more preventive measures [3]. More recently, an expansion in methodology has led to a number of studies using mathematical modeling [4,6,7] and simulation [5] to investigate the spread of MRSA within hospitals. These computational models allow researchers to evaluate potential solutions in a virtual environment so that hospital administrators can make informed decisions concerning infection control policy. However, mathematical models have limitations as well, as they are driven by dynamic equations that represent the macroscopic behavior of the system. Even when properly calibrated, these models lack realism as they fail to depict the low-level interactions that drive the entire system. These interactions are more naturally represented by agent-based modeling [8], a more recent approach, which is used to develop a software package to further investigate this problem.

Agent-based modeling (ABM) is a powerful technique that seeks to generate emergent characteristics from simple, rule-based individual behavior. In other words, the goal of ABM is to determine whether or not macroscopic trends can be generated from microscopic actions. This technique is used to define agents in a hospital, specifically patients, nurses, physicians, and visitors, that interact with each other throughout the simulation period. The interactions between agents serve as the source of transmission dynamics within the hospital. Discrete event simulation (DES) is used to propagate the interactions between the agents and serve as an interface to collect data for various configurations of hospital operations, such as the implementation of specific ICMs. The simulation is stochastic, implying there are many events that are determined through the use of pseudo-random number generation. The stochastic nature of the simulation requires multiple replications of each scenario to be executed, and thus Monte Carlo methods are also incorporated into the design of the software.

DES typically consists of three design methodologies: time stepped DES, event oriented DES, and process oriented DES. Time stepped DES propagates time using a fixed time step until the simulation time of a scheduled event has been reached, at which point the event is processed and time is advanced further. A significant disadvantage of time stepped DES is that if the events are distant in time, the simulation could propagate a long time without processing any events, which is an inefficient use of computational resources. Event oriented DES advances time to discrete simulation times at which an event is scheduled to occur, processing the scheduled event and then proceeding to the next scheduled event time. This methodology leads to a serial processing of events, which is simpler to implement, but still inefficient. Process oriented DES operates in a slightly different way, where each simulation component is modeled as a process that executes until the simulation has reached a terminating condition. Process oriented DES also advances to discrete simulation event times, but the execution of the simulation occurs as a series of parallel processes. The process oriented DES methodology is

becoming the most common technique used in simulation currently, and thus is the method of choice for this software project.

Implementation

In order to implement the agent-based model, each agent is defined in terms of its characteristics and behavior. This type of modeling is supported best by object-oriented programming (OOP), in which object classes are defined with inherent characteristics and methods. The simulation package is developed in Python, a dynamic object-oriented programming language [9]. In addition to basic Python, the NumPy, SciPy, and SimPy modules are also useful resources for building the software package. NumPy is a multi-dimensional array-based module that contains a large number of operations for arrays. SciPy is a module used for scientific computation tasks, which provides random number generation functions. SimPy consists of process oriented DES classes and methods which are used to develop the simulation architecture for the software.

There are two major object types in SimPy: processes and resources. The interactions between processes and resources are simulated through the use of a scheduler, which advances to discrete points in time to handle specific events. Processes can be used to model many real world objects that are progressing through a system. In SimPy, processes advance through a series of active and passive states defined by their process execution method (PEM) to represent the passage of time. There are a number of ways to start and stop processes during their execution. The first series of methods are yield statements, which cause a process to wait in a passive state until a certain criteria is met, such as a fixed passage of time or until a certain resource has been acquired or released. The second type of process control uses interruptions to awaken a process that is currently waiting. The process that is interrupted can determine the source of the interruption, and is then able to interact with that process in some way. The third method of process control uses events to signal waiting processes. This method can be implemented in two variations. The first technique is for all processes to be awakened that are waiting on the event once the event has been signaled. The second technique is that processes enter a queue for that specific event, and are awakened in a first-in first-out (FIFO) manner as the event is signaled. All of these techniques are used in the software to control the behavior of the agents as they interact with each other.

The agents in the simulation are all represented as processes, including patients, HCWs, visitors, and even hospitals. The only allowable interactions are between patients and HCWs, and patients and their visitors. Interactions between HCWs are not modeled at this time. All agents in the simulation are generated by a source agent, which varies in its operation depending on the type of agent being generated. Patients are generated continuously throughout the simulation as space becomes available in the waiting room of the hospital. This technique allows patient agents to be generated ‘as needed’ so as to prevent an *a priori* determination of the number of patients required to seed the simulation. HCWs are generated at the beginning of the simulation, as specified by

parameters. A fixed number of visitors are generated each day that each visit a single patient in the hospital at random.

There are three classes of resources in SimPy: resources, levels, and stores, two of which are used thus far in the software. Resources can have a finite or infinite capacity and are requested one unit at a time by processes. Hospital rooms, specifically beds, are modeled as resources that are requested by patients as they enter the hospital. Stores are finite capacity resources that can actually contain processes themselves, which can be requested by other processes in single or multiple quantities. Nurse and physician staffs are modeled as stores, where patients can request either type of HCW for a visit, before returning them so that they become available to other patients.

The transmission of MRSA can occur in one of three ways:

1. A newly admitted patient transmits the bacteria to an HCW,
2. a colonized HCW transmits the bacteria to the patient, or
3. a colonized visitor transmits the bacteria directly to the patient

The environment does not contribute to the likelihood of transmission or acquisition at this time. The transmission of MRSA between agents is determined stochastically, based on the risk level of the patient and the behavior of the HCWs that visit. Once colonized, a patient remains colonized until the patient either develops an infection or completes a decolonization regimen. The patient can only begin the decolonization or treatment process once the state of the patient has been determined by an HCW. The colonized state of the patient can be determined through a screening test whereas the infected state is ascertained by visual confirmation. An HCW can only become colonized through direct contact with a patient. A colonized HCW can become decolonized upon the occurrence of its next hand hygiene activity such as hand washing or using alcohol-based gels. The probability of an HCW agent washing its hands is based on its own hand hygiene compliance, factoring in the risk level and isolation status of the patient. Agent interactions and state transitions are summarized in Figure 1.

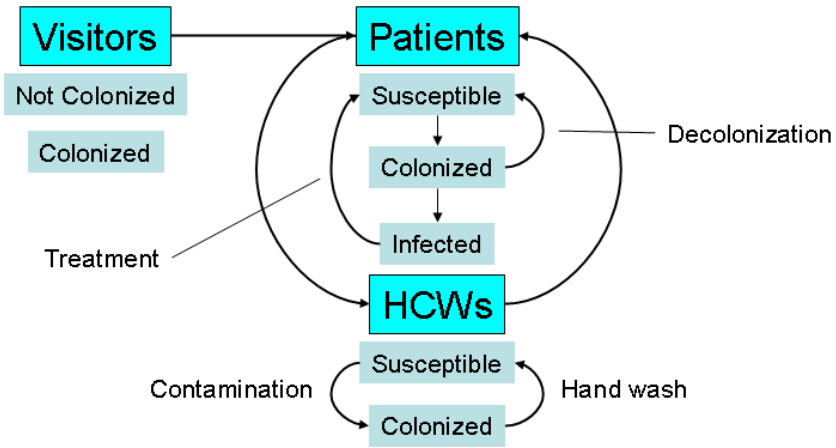

Figure 1: Agent Interactions and State Transitions

The general operation of the software includes three major stages: initialization, simulation, and output of results. During the initialization phase, all of the required modules are loaded into the simulation environment, including the agent class definitions, NumPy, SimPy, and SciPy modules. In addition, simulation parameters are defined such as the number of days to be simulated and the number of Monte Carlo replications to be executed. The hospital is also defined in this phase with a specified number of single and double rooms along with the infection control policy. Finally, the nurse and physician staffs are created for the hospital.

During the simulation phase, all of the processes are activated and proceed through their process execution methods. Patient agents enter the hospital and request a bed. When a bed becomes available, the patient is admitted and possibly screened if prescribed by the hospital infection control policy. If a bed is not yet available, the patient occupies a space (if available) in the hospital waiting room. Patient lengths of stay and the required number of visits per day are determine stochastically once the patient is admitted. Patients are visited each day by nurse and physician agents, and sometimes visitor agents, if they are fortunate enough to have friends and family. If active surveillance is enforced, the patient may be screened periodically for colonization. Once the screening test results have returned, the patient may receive treatment based on the results. If the patient tests positive for colonization or shows signs of an infection, the patient may undergo decolonization or be moved to isolation. During each visit, MRSA can be transmitted to or from the patient. If a nurse or physician visits the patient, they have the opportunity to wash their hands once the visit is complete. For each patient that acquires an infection, their stay is extended by a randomly generated period of time. Otherwise, that patient is released at the end of their predetermined period of stay and another patient is admitted.

At the end of each replication, statistics are accumulated before moving on to the next replication. Once all replications have been simulated, an auxiliary function is called, which prints the results of the simulation. The output displays three categories of information, including the simulation parameters as well as the infection control policy of the hospital and the simulation results averaged over the number of replications. The simulation parameters consist of the number of simulated days and replications as well as the size of the hospital and medical staffs. The policy is summarized in terms of which measures are taken in addition to any parameters involved with a particular measure, such as the frequency of patient screening, delay in the return of screening test results, or the decolonization period for colonized and infected patients. The simulation results present a number of statistics related to the execution of the simulation, many of which are averaged over the number of replications. The first set of statistics summarize the population statistics within the hospital, such as the total number of patients, the number of patients discharged, and the average length of stay. The next set of statistics display information related to the implemented ICMs, such as the number of patient screening or the number of patients that completed the decolonization process. The last series of statistics relate to the infection metrics, which summarize the spread of infection within the hospital. Finally, there are two measures output intended to track execution time, including the number of random number generator calls and the execution time itself. A sample output of the software is shown in Figure 2.

```
In [39]: reload(HSim)
Simulation Parameters

Simulated Hospital Days: 100
Number of Replications: 5
# of Hospital Rooms (Single/Double): 10/20
# of Nurses: 10
# of Physicians: 5

Infection Control Policy

Nurse Hand Hygiene Compliance (Low/High Risk Patient): 0.5/0.8
Physician Hand Hygiene Compliance (Low/High Risk Patient): 0.5/0.8
Patient Isolation: False
Patient Screening on Admission: True
Patient Screening During Stay: Every 5 days
Patient Screening Results Returned in: 1 day
Patient Cohorting: False
Nurse Cohorting: False
Decolonization: True
Decolonization Period (Colonization/Infection): 5 days/5 days

Simulation Results

Random number generator calls: 357199

Mean # of Patients: 1020.4
Mean # of Patients Discharged: 931.6
Mean # of Patient Days: 5000
Mean Length of Stay: 5.03308341993 days

Mean # of Patient Screenings: 1436.0
Mean # of Patients Decolonized: 35.4

Mean # of Patients Colonized: 117.0
Mean # of Colonized Patients Admitted: 97.6
Mean # of Patients Colonized By Visitors: 6.8
Mean Successful Introduction Rate: 12.6
Mean # of Patients Infected: 3.0

Mean Ward Prevalence: 0.974
Mean Colonized Patient Days: 0.08092
Mean Attack Rate: 0.0137093615354
Basic Reproduction Number, R0: 0.129098360656

42.5469999313  seconds have elapsed
```

Figure 2: Sample Simulation Output

Validation and Testing

There are a number of standard metrics from the literature that are used to evaluate the effectiveness of infection control procedures. The first of these is the *successful introduction rate*, which is the number of secondary MRSA cases arising from transmission within the hospital. This metric gives a clear depiction of the susceptibility of the hospital to outbreaks. Another important metric is *mean ward prevalence*, which is the percentage of hospital days on which at least one colonized patient was present. This metric describes the degree to which MRSA is present within the hospital, which correlates with the ability to eliminate the bacteria completely. The percentage of *colonized patient days* represents the proportion of days spent as a colonized or infected patient in the hospital. This metric is a good representation of quality, as low proportions indicate not only a low incidence of cross-transmission, but a quick response to those patients who have become colonized or infected. The *attack rate* is calculated as the ratio of MRSA transmissions to uncolonized patient days. This metric represents the number of uncolonized patient days between a secondary case of MRSA. Finally, the *basic reproduction number*, R_0, is a key indicator of whether an outbreak will occur. R_0 is the mean number of secondary cases that results from a single source patient, being one who was admitted at a colonized state. Typically, if $R_0 > 1$, then an outbreak is likely to occur

within the hospital, as the average colonized patient will transmit MRSA through an HCW to at least one other patient.

As a preliminary test, the effects of hand hygiene compliance were investigated with respect to successful introduction rate and R_0. HCW hand hygiene compliance was varied from 0% to 100% in 10% increments. The simulation was executed for a hospital with 10 single and 20 double rooms for a period of 100 days. There were no ICMs implemented to test the susceptibility of the hospital to an outbreak. The results of the study are shown in the figure below, averaged over 5 simulation replications:

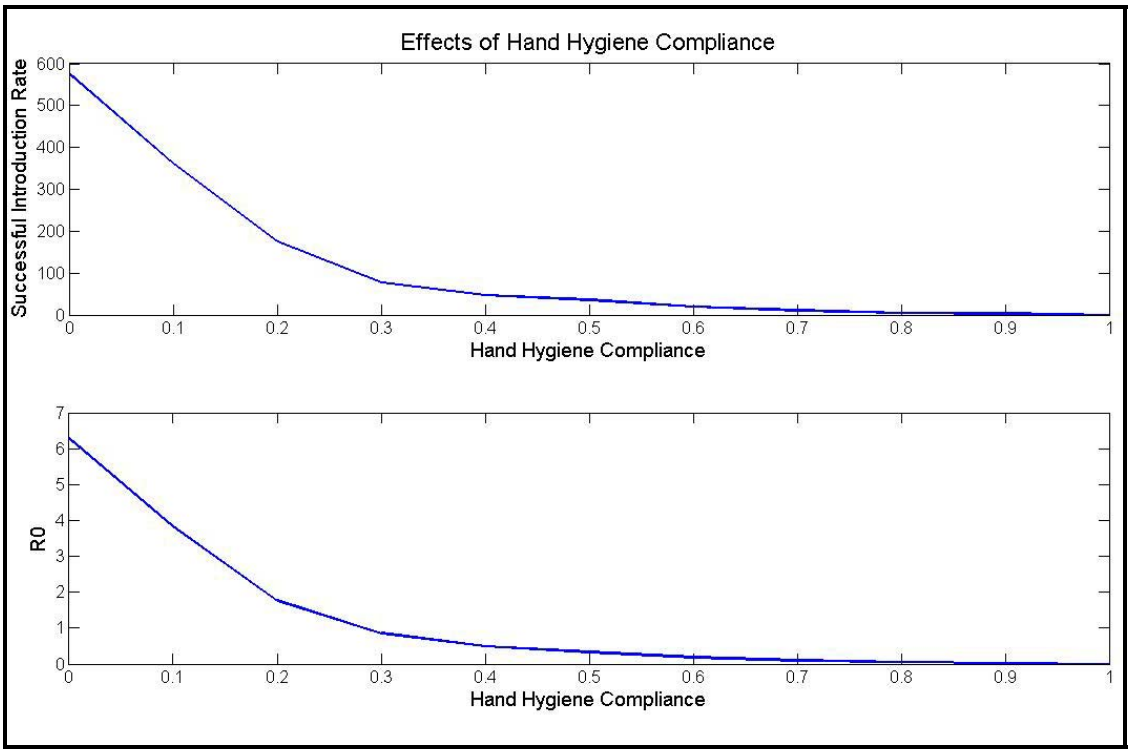

Figure 3: Results of Hand Hygiene Compliance Study

It is clear from the plots in Figure 3 that 30% compliance appears to be a threshold value. For compliance rates below 30%, a significant MRSA outbreak occurs in the hospital. At higher rates of compliance, the spread of MRSA appears to be contained. It is no coincidence that a hand hygiene compliance of 30% corresponds approximately to an R_0 of unity. It is also evident from the plots that increasing hand hygiene compliance displays the law of diminishing returns, requiring dramatic increases in compliance to reduce transmission significantly further, as concluded from the literature (Figures 4 [4] and 5 [6]).

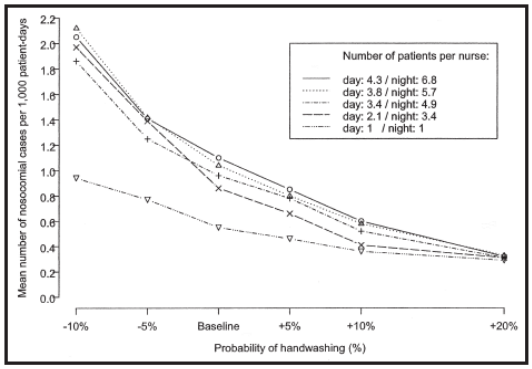

Figure 4: Effects of hand washing on the number of secondary cases per 1000 patient days

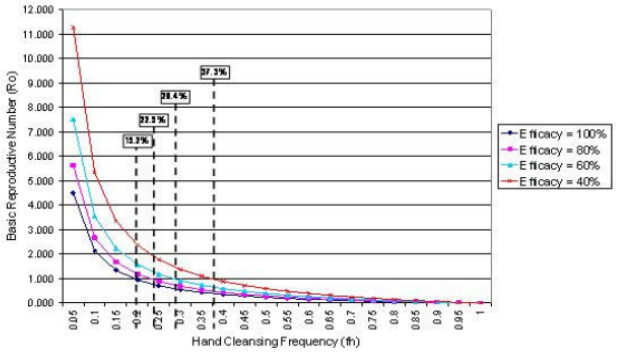

Figure 5: Effects of hand washing frequency efficacy on the basic reproduction number, R_0

It is clear at this point that a crucial issue with this software and with agent-based modeling and simulation (ABMS) in general is simulation execution time. There is a relatively small amount of code at compile time, but this increases many times over at execution time, due to the many process instances executing the same PEMs during the simulation. Therefore, only small examples are capable of running on single processors in a reasonable amount of execution time. As verification, the following two examples were tested on a personal laptop and a single processor on the Genome Cluster at the University of Maryland. The Genome Cluster currently employs 16 processors and 64GB of RAM. At this point, the implementation of the software is entirely serial, thus the computing power of the Genome Cluster has not yet been directed at this issue.

	Small Example	Large Example
Simulation Days	100	1000
Hospital Capacity (beds)	15	500
Staff (nurses/physicians)	5 / 3	50 / 20
Infection Control	Screening/Decolonization	Screening/Decolonization
Replications	5	50
PC Execution Time	13.1 seconds	13.2 hours
Genome Execution Time	9.2 seconds	9.2 hours

Figure 6: Execution Time Summary

Future Work

The first priority of this ongoing work will be to continue the implementation of the infection control measures, beginning with patient isolation. Now that patient screening has been implemented and HCWs are capable of detecting colonized and infected patients, measures can be taken to control transmission throughout the hospital. In addition to the not yet implemented infection control measures, there will be some expansion on the current capabilities, including the implementation of a hand hygiene efficacy parameter and probabilistic patient screening.

The second priority will be to improve the simulation execution time, through the examination of code efficiency and the investigation of a parallel implementation, which will be capable of taking advantage of the multi-processor Genome computing cluster at the University of Maryland. The parallel implementation will be developed using Parallel Python (PP), an additional Python module that allows for simple parallelization of code. The parallelized version of the software will run separate simulation replications on multiple processors and aggregate the simulation results.

The third major focus will involve further validation and testing of the software. This process will include additional trend matching as described in the literature and new avenues of exploration. In order to make testing more feasible, a parameter input system will be designed, so that test runs can be executed with a higher degree of automation. If time allows, a graphical user interface (GUI) will be developed to facilitate this process. In addition, output files will be generated so that results will not require manual recording. MATLAB may be used to perform output analysis and visualization.

Project Schedule

The original project schedule is shown in Figure 7. At this stage of the project, it appears that my progress has kept to the schedule fairly well. The project definition state has been completed, although we are planning to meet with UMMC again early next year. The software development phase has progressed significantly, and as a result, some additional features will be implemented in the future. The class definitions, architecture, Monte Carlo, and metric tracking aspects are already implemented, although they may be expanded further. Preliminary testing has already begun, but the majority of this phase will be implemented in the future. Looking forward, it appears as if the software and analysis milestones will be delayed somewhat, but no more than 2-4 weeks as there are additional implementation aspects that have been added to the original proposal, such as the parallel implementation and expanded infection control measures. This delay will not interfere with the final documentation and presentation tasks.

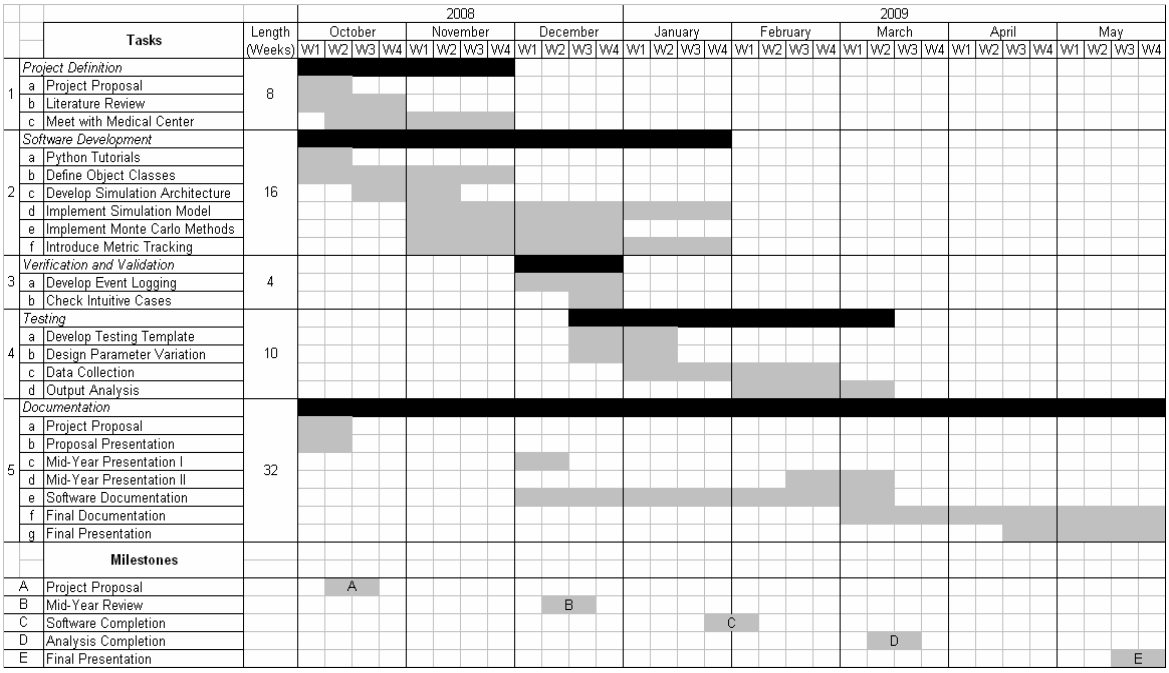

Figure 7: Original Project Schedule

Acknowledgements

Special thanks to Dr. Harold Standiford, Medical Director of Infection Control and his staff at the University of Maryland Medical Center, Baltimore, MD; Dr. Edward Wasil, American University; Dr. Catherine Dibble, Aiki Labs; and Carter Price for their time, suggestions, and expertise.

References

1. Committee to Reduce Infection Deaths. www.hospitalinfection.org
2. McCaughey B, 14 Aug 2008. "Hospital Infections: Unacceptable and Preventable". Wall Street Journal, pp. A11.
3. Griffin FA, 2007. Reducing Methicillin-Resistant *Staphylococcus Aureus* Infections. The Joint Commission Journal on Quality and Patient Safety (Vol. 33, No. 12), pp. 726-731.
4. Cooper BS, Medley GF, Scott GM, 1999. Preliminary analysis of the transmission dynamics of nosocomial infections: stochastic and management effects. Journal of Hospital Infection, Vol. 43, pp. 131-147.
5. Raboud J, Saskin R, Simor A, Loeb M, Green K, Low D, McGeer A, 2003. Modeling Transmission of Methicillin-Resistant *Staphylococcus Aureus* Among Patients Admitted To A Hospital. Infection Control and Hospital Epidemiology, Vol. 26 (7), pp. 607-614.
6. McBryde ES, Pettitt AN, McElwain DLS, 2007. A stochastic mathematical model of methicillin resistant *Staphylococcus aureus* transmission in an intensive care unit: Predicting the impact of interventions. Journal of Theoretical Biology, Vol. 245, pp. 470-481.
7. Beggs CB, Shepherd SJ, Kerr KG, 2008. Increasing the frequency of hand washing by healthcare workers does not lead to commensurate reductions in staphylococcal infection in a hospital ward
8. Macal CM and North MJ. Agent-Based Modeling and Simulation: Desktop ABMS. INFORMS Winter Simulation Conference, December 12, 2007. Washington, D.C.
9. Vignaux T, Muller K, and Helmbold B, 2007. The SimPy Manual. Available at: http://simpy.sourceforge.net/SimPyDocs/Manual.html