DEVELOPING THE OBJECT-ORIENTED DYNAMIC MOSQUITO SIMULATION MODEL (DYMSIM)

By

SONIA SEN

A Thesis Submitted to The Honors College

In Partial Fulfillment of the Bachelors degree
With Honors in

Geography

THE UNIVERSITY OF ARIZONA
DECEMBER 2014

Approved By:

[illegible]

Dr. Andrew C. Comrie
School of Geography and Development

ACKNOWLEDGMENTS

I would like to first and foremost thank Dr. Andrew Comrie for accepting a wide-eyed freshman into his research group four years ago. Andrew has always supported my academic (and non-academic) endeavors throughout my whole college career and for that I am truly grateful. Furthermore, I would like to thank Dr. Cory Morin for his friendship and commitment in making this whole project possible. Additionally, I would like to thank Mona Arora, for her continued support and guidance throughout the past few years. The Arizona NASA Space Grant for funding me for two years of completing this work and specifically Susan Brew and Dr. Chandra Holifield-Collins for their positivity and encouragement. Of course, my family for their motivational talks and unending reinforcement during the past two decades. Finally, to my friends who have become like family, for listening to me talk about mosquitoes for the past four years.

TABLE OF CONTENTS

ABSTRACT

Projected changes in climate pose a variety of potentially harmful outcomes for the current Earth system, including the rates of vector-borne infectious diseases. The Dynamic Mosquito Simulation Model (DyMSiM) was previously developed to model and predict the rates of mosquito-borne infectious diseases based on precipitation and temperature data. Based in Stella, a development software program, the model's current setup limits functionality and speed whereas an object-oriented version of the DyMSiM software was identified as potentially alleviating those constraints. The conversion into an object-oriented model offered the freedom of access to users, customizability, and more specific outputs among other features. Java was used in programming the model, which consisted of six different classes: Model, Container, EggBatch, LarvaeBatch, PupaeBatch, and Adult. The Java model code enabled climate data (precipitation and temperature) to be entered via an Excel spreadsheet. The egg, larval, and pupal stages of the mosquito lifecycle were treated as "batches" in which a cohort of multiple simulated developing mosquitoes go through the same process together. This method was not used, however, for the Adult stage of the mosquito, as Adults were instantiated as their own respective objects in the model. The model was validated against the Stella version using example data from San Juan, Puerto Rico in 2010 and 2011. Resulting model outputs of infectious mosquitoes were found to be within an order of magnitude as the Stella model. The development of this object-oriented model has created the foundation upon which many new specifications and additional features to the model can be made to enhance the study of mosquito-borne diseases around the world.

INTRODUCTION

Vector-borne diseases account for 17% of estimated global incidences of infectious disease and are of particular interest to climate and epidemiological researchers (WHO 2014). Moreover, due to the assortment of variables affecting them, vector-borne diseases are often difficult to predict and model (CDC 2014). While it is known that climate plays a significant role in the development and transmission of these diseases, specifically in regards to temperature and rainfall, the projected climate changes around the world provide an additional layer of complexity in understanding vector-borne disease's ecology and pathology. A variety of stakeholders, including public health workers, entomologists, policymakers and citizens of areas endemic to a particular vector-borne disease, have a need for historical data to make predictions about anticipated rates of infectious diseases to reduce infection and mortality rates in the population. However, a widespread lack of suitable data makes empirical and statistical modeling difficult or impossible in most locations. Instead, dynamic mathematical simulations are needed to fill this need (Morin, Comrie 2010).

Dynamic Mosquito Simulation Model

The Dynamic Mosquito Simulation Model (DyMSiM) utilizes climate data in order to simulate mosquito populations over a given area (Morin, Comrie 2010). DyMSiM allows users to input the two climate variables that have the greatest effect on mosquito populations, namely precipitation and temperature, and observe the outputs of mosquito population size and disease incidence (Figure 1). Figure 1 below also shows the other components of the model, including land cover, containers for which the mosquitoes to breed, daylight and evaporation. Specifying precipitation and temperature effects allows simulation of the mosquito lifecycle, including a specified disease carried by the mosquito. By using past climate and vector data to validate the model, DyMSiM allows a variety of stakeholders to predict future populations of mosquitoes without local mosquito data, and therefore assess the potential for disease outbreaks. DyMSiM was originally developed to simulate *Culex quinquefasciatus*, which is a carrier for West Nile Virus in the southern United States (Morin, Comrie 2010). Though this was the original specification for the model, DyMSiM can and has been used to study a variety of different mosquito vectors and diseases in different environmental settings. Along with West Nile Virus, DyMSiM has simulated Dengue fever with the *Aedes aegypti* mosquito in the Caribbean region (Morin, Comrie, Ernst 2013).

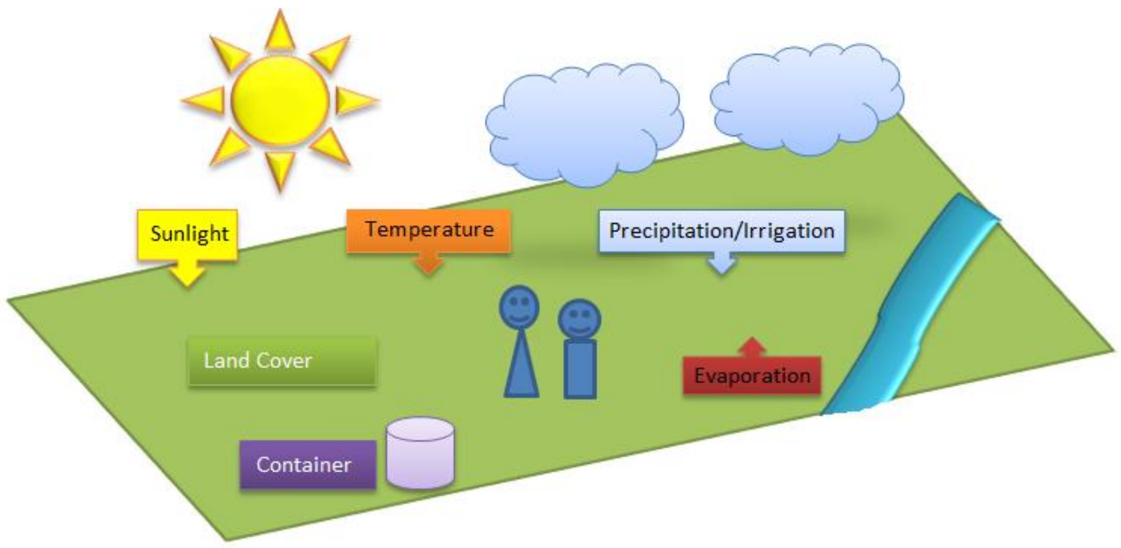

Figure 1: Graphical depiction of the environmental inputs of DyMSiM

Application – Dengue Fever in Puerto Rico

The object-oriented model was validated using *Aedes aegypti* mosquito vector data and climate data from San Juan, Puerto Rico. This set of data was primarily chosen because it had already been validated by existing Stella DyMSiM software runs. This vector transmits Dengue fever, which has been endemic to the island for the past 50 years. There have been four epidemics in the past 20 years on the island, each having caused tens of thousands of people to become infected with the disease. DyMSiM can be used to improve understanding of the disease's spatial and temporal characteristics on the island and make for more informed public health decisions for prevention. Additionally, studying Dengue fever and the potential effects of climate change on the disease and its vector are particularly interesting because of the projected changes in climate for the island. Predictions for Puerto Rico include sea level rise coupled with warmer air and water temperatures, which could create potentially favorable environmental conditions for the *Aedes* mosquito species.

Motivation

The reasoning of converting DyMSiM from the existing Stella modeling software to a Java-based model was twofold:

1) Increase the usability so that the model was not limited to users who had access to Stella
2) Allow the model to become highly customizable beyond the constraints posed by the Stella modeling software.

Converting DyMSiM to an open-source and widely used software platform (i.e. Java) would not only provide improved understanding and control of the processes being simulated, but also create a better comprehension for users by making the model an easier product to use. The current dependency on Stella software, means that not only do many potential users not have access to the software, but also the intricacies of running the model requires a deep understanding of how the specific DyMSiM model is implemented in Stella. While Stella is a popular tool used by many academic and industry climate modelers alike, there is still a major barrier in its prohibitive licensing costs for some. When DyMSiM on Stella was made openly available online, a multitude of users ranging from fellow university affiliates to governmental health workers were interested in downloading and using the software. However, because of the model's dependence on Stella, there was a somewhat laborious process in familiarizing the users in how to use the model, if they had the software to begin with. Potential outcomes and new features of an object-oriented model, many of which were realized in this honors thesis and others which are discussed in the Future Work section, was another main driving factor in the development. These additional features of the model include: increased usability by not limiting the model to proprietary software, easier customization of inputs and outputs, the potential for making the model web-based, and the opportunity to enable a graphical user interface for educational purposes. All of these factors made a compelling case for the creation of the object-oriented model and the pursuance of this honors thesis project.

METHODOLOGY

Java

Java was the programming language of choice to recreate DyMSiM as an object-oriented piece of software. The language is one of the most widely utilized programming languages across the globe, ensuring vast easily available technical support as well as ensuring a broad user audience. Specifically, Java was developed and popularized because of its "Write Once, Run Anywhere" (WORA) catchphrase, signifying that Java code, regardless of where and how it is written, can be run on any Java-enabled platform. The issue of technical interoperability was of particular importance for the development of this model because of the aforementioned hindrance to usability by having DyMSiM only available through the expensive and complicated Stella software. Developing in Java, which can eventually be programmed to have a simple graphical user interface, allows for the endless possibilities in how it can be tailored to run. Furthermore, Java is also extremely popular for its usage in web-based applications and mobile devices, a consideration that is valuable for future plans and expansions of DyMSiM.

Stella Diagrams vs. UML Diagrams

Below are the logic diagrams from the original Stella- based DyMSiM model. The intricate network of interconnectedness and dependability attests to the major knowledge barrier that DyMSiM on Stella posed to potential users of the model (Figures 2, 3).

Figure 4 is the basic Unified Modeling Language (UML) diagram of the object-oriented DyMSiM. UML is the *de facto* standard for visualizing the design of a software system, used especially in the setting of object-oriented designs. While Stella was valuable in initial development of the underlying science for DyMSiM, the rewriting of the established model enables a simple and efficient programming structure that is also much easier to understand.

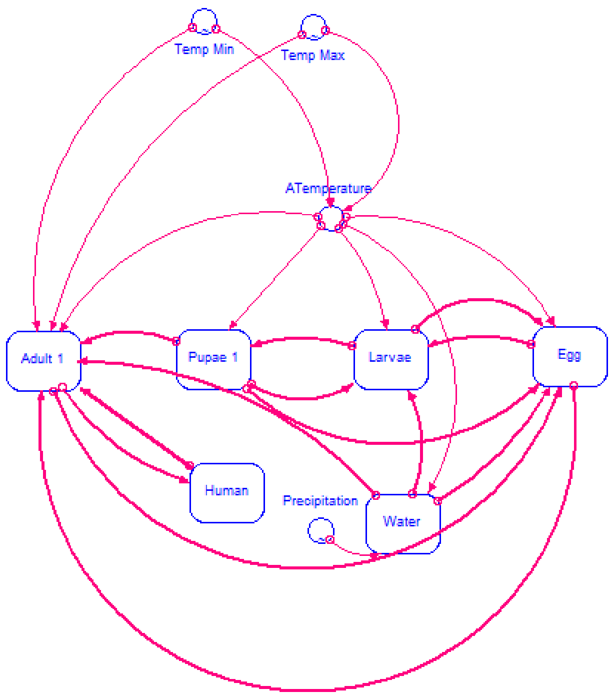

Figure 2: Stella representation of DyMSiM

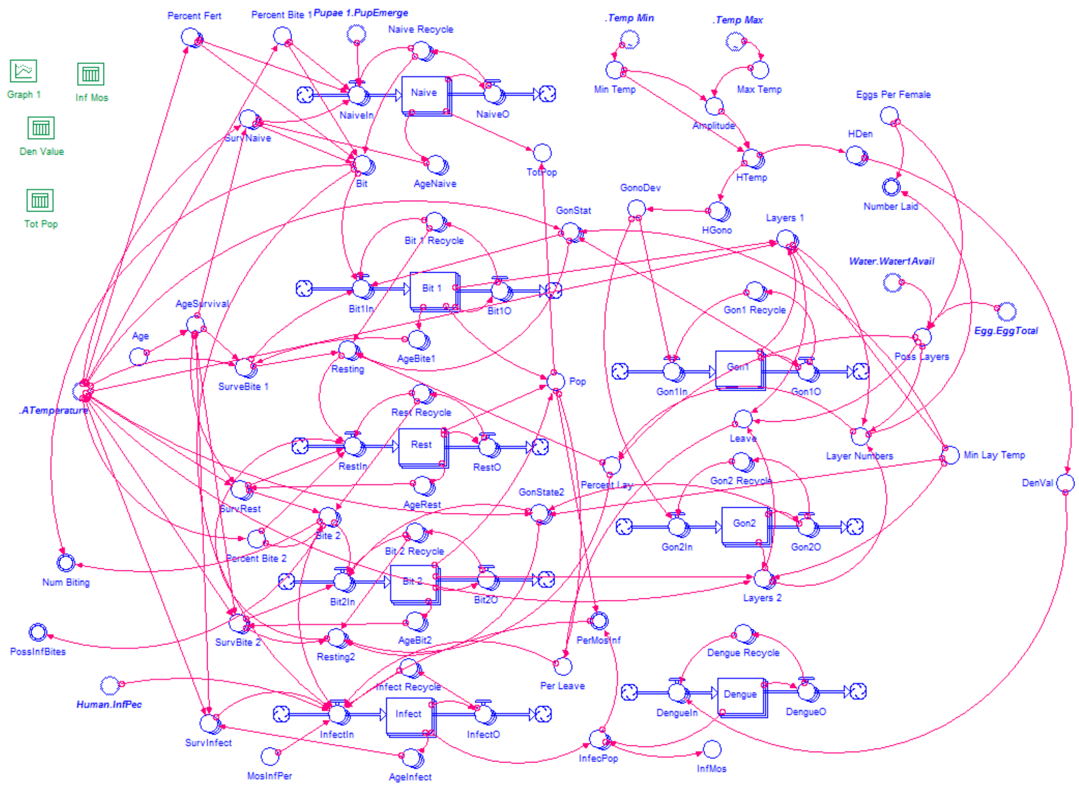

Figure 3: Stella representation of the Adult lifecycle stage

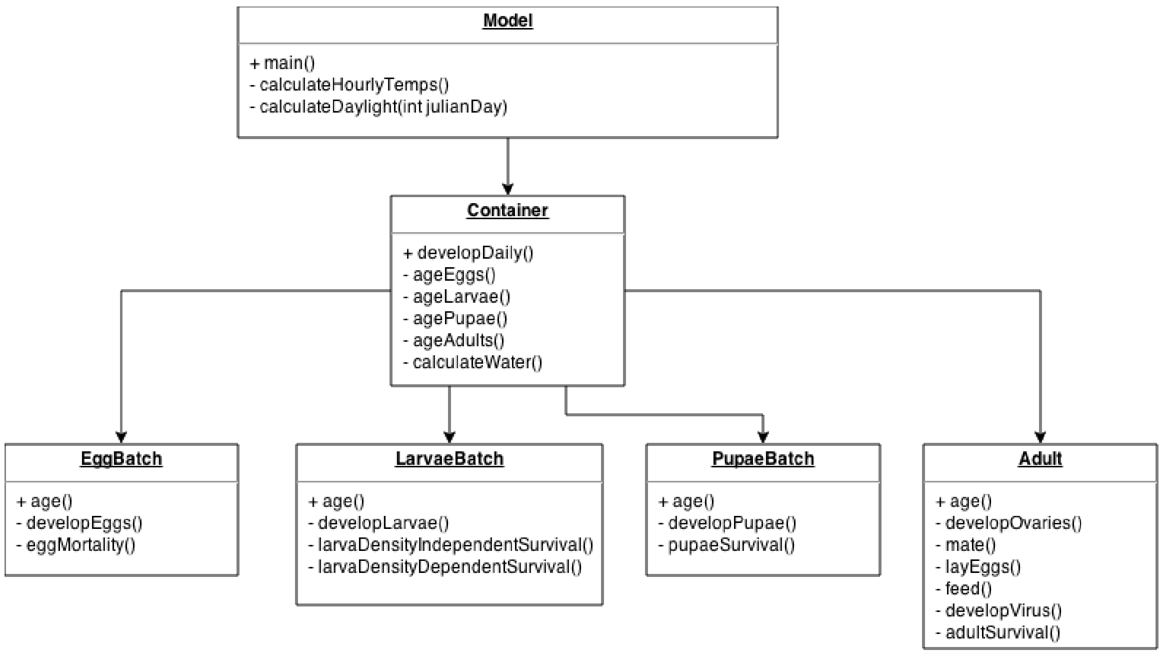

Figure 4: UML diagram of the object-oriented DyMSiM

Model and Container

The foundational object components of the DyMSiM are that of the Model object and the Container object.

The Model object is the overarching “home” of DyMSiM. It is the part of the Java application that is actually being “run” and the only class that has any interaction with the user. Model takes in the given inputs of temperature and precipitation data that are entered via a Microsoft Excel spreadsheet for ease of use. The input data consists of temperature and precipitation for one to a few years in a specified area. The length of time for which the Model should run is also specified and run in this section.

The input of temperature for the model is given as the observed (or projected) minimum and maximum temperatures for the day. It is important to note the the delay in temperature variations on a diurnal basis. In particular, while solar noon is the point at which the sun is at its highest zenith angle and the earth is receiving the maximum amount of solar radiation, this point almost never coincides with the hottest part of the day. This is because there is a lag effect that as the land surface and overlying air warm through the day, leading to the highest temperatures in mid-afternoon. In order to simulate this diurnal variation in temperature, while also attaining hourly temperatures that are necessary for the model, but not commonly recorded, a sine wave calculation was used. So the Model object, while reading in the maximum and minimum temperatures provided by the user, is also calculating hourly temperature values based on this sine variation. This improves model calculations by providing diurnal variability for the mosquitoes lifecycle development.

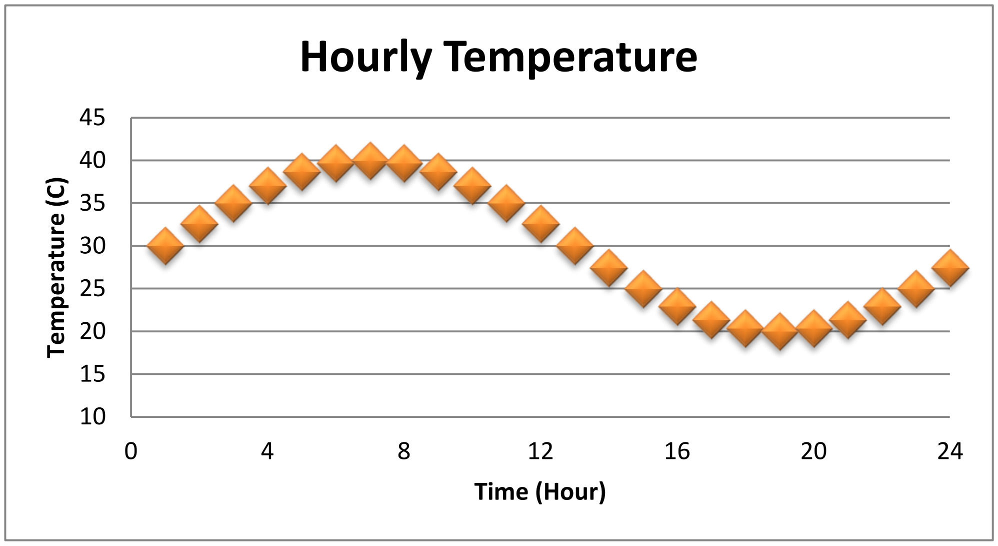

Figure 5: A sample of simulated diurnal hourly temperature variation given a maximum temperature of 40°C and minimum temperature of 20°C

The Container object is where the mosquito lifecycle is simulated. With specifications passed from the Model including location, temperature, precipitation, and area of the object, the Container represents a specific enclosure where mosquitoes may breed (e.g., drainage canals or roof gutters).   Each Container object keeps track of the different stages of mosquito populations living in it: Eggs, Larvae, Pupae and Adult, all at a daily resolution (Figure 6).

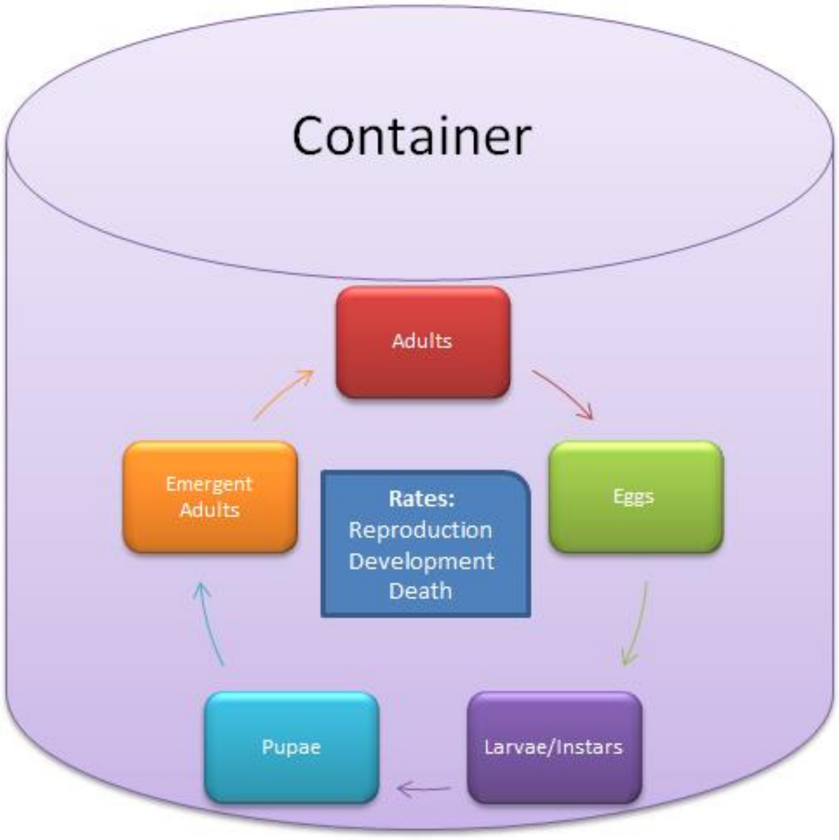

Figure 6: Graphical depiction of the Container object showing the lifecycle stages of the mosquito.

Container, using the specified climate data, advances or ages the populations of the mosquitoes accordingly on a daily basis. The actual "aging" process, however, is done in each respective object associated with the Container, i.e., an egg batch in the Container will call upon the methods written in EggBatch to age and develop.

Separating the Container object from the Model object allows for future scalability of the model. It achieves this by allowing a Model to contain many containers, enabling the model to run over a spatially large and diverse area with different types of containers. Within the span of backyard, for instance, containers ranging from pools to dog bowls to toy buckets to potted plants can all present perfect breeding container grounds for mosquitoes. Allowing the model to simulate multiple containers allows this more realistic simulation of typical habitats.

Eggs

The EggBatch object represents the single laying of eggs by a female mosquito, which can sometime consist of 100 – 200 eggs at a time. EggBatch also has a rescue effect where if there are no more EggBatches due to death in the model, 100 eggs will automatically be added, so that there is no extinction of the mosquitoes. Treating the hundreds of eggs that exist in a Container as "batches" eliminates unnecessary memory usage, especially since each egg in a batch has the same survival and development rates as the other eggs in its batch. The Container will call upon the age() method in the EggBatch class for every egg batch it comprises of, which then advances the population size according to the day's climate data.

There are two calculations that advance the EggBatch. The Egg Development signifies how the Egg Batch is progressing in its growth towards becoming a Larvae Batch. Egg Survival determines the percentage of whether or not the Egg Batch will survive on the given day. Variations in climate conditions affect the daily survival and development rates of the egg population of an Egg Batch as shown in Tables 1 and 2:

Table 1: Egg Development

Temperature Range	**Equation/Constant**
$T < 14.5$, $T > 36$	0
$14.5 \leq T \leq 31$	$0.0012*T - 0.0171$
$31 < T \leq 35$	$-0.0002*T + 0.0268$
$35 < T \leq 36$	$-0.0198*T + 0.7128$

Table 2: Egg Survival

Temperature Range	Equation/Constant
$T \leq 0, T \geq 38$	0
$0 < T < 15$	$0.071*T - .071$
$15 \leq T < 32$	0.98
$32 \leq T < 38$	$-0.0141*T^2 + 0.8278*T - 11.096$

Larvae

The larval stage of development for mosquitoes occurs after the larvae hatch out of the laid eggs, given that the appropriate hydrological conditions are met. Eggs will remain dormant until there is enough water for the larvae to hatch. The Larvae object is also treated as a batch (Tables 3, 4).

Table 3: Larvae Development

Temperature Range	Equation/Constant
$T \leq 14, T > 38$	0
$14 < T < 36$	$0.0076*\ln(T)-0.0192$
$36 \leq T \leq 38$	$-0.0027*T+0.1041$

There is an additional consideration for Larvae survival depending on the density of the populations in the Container. This additional calculation is to account for the fact that when space resources are low, the larger, more developed Pupae populations in the Container will take more of the resources from the Larvae, resulting in their death. The carrying capacity of the container signifies the amount of Larvae per Pupae per milliliter of water (Table 5). Currently, according to literature research, the DyMSiM has specified a 0.5 larvae/pupae/ml carrying capacity (Morin, Comrie, 2010).

Table 4: Larvae Density Independent Survival

Temperature Range	**Equation/Constant**
T ≤ 4, T ≥ 38	0
4 < T < 15	0.0876*T - .3502
15 ≤ T < 35	0.98
35 ≤ T < 38	-0.1863*T+7.4581

Table 5: Larvae Density Dependent Survival

Density Range	**Equation/Constant**
< Carrying Capacity	1
> Carrying Capacity	(Carry Cap*Habitable Water)/(Lar+Pup Pop)

Pupae

The third stage of mosquito development is of the pupae. This is the final stage before the mosquitoes become adults. This stage in DyMSiM is fairly straightforward with temperature and precipitation determining the development and survival rates. As with Egg and Larvae, the Pupae object is treated as a batch too (Tables 6, 7).

Table 6: Pupae Development

Temperature Range	**Equation/Constant**
T ≤ 12, T ≥ 39	0
12 < T < 32	.0014*T-.0155
32 ≤ T < 39	-0.0016*T^2+0.1061*T-1.7746

Table 7: Pupae Survival

Temperature Range	Equation/Constant
$T \leq 4, T > 39$	0
$4 < T < 15$	$-0.0068*T^2 + 0.2201*T - 0.7794$
$15 \leq T < 33$	0.99
$33 \leq T \leq 39$	$-0.1585*T+6.2021$

Adult

The final stage of the mosquito lifecycle is that of the Adult stage. The Adult object in DyMSiM, however, is not treated as a batch. Rather, the Adult objects in the model uniquely represent the presence of a single mosquito in the simulation. While the calculation of the survival rate is similar, the Adult also has different developmental processes as compared to Egg, Larvae and Pupae (Figure 8).

Biologically speaking, DyMSiM does not simulate the whole mosquito population in the real world, rather only the females. This is because male mosquitoes do not feed on blood; female mosquitoes bite to receive blood meals, and thus spread disease, in order to ovulate. This stage is the most important for researchers interested in the public health outcomes of DyMSiM because while the other steps are necessary to achieve adulthood, it is only adult females that may become infectious and therefore spread the disease (Tables 8, 9, 10, 11, 12).

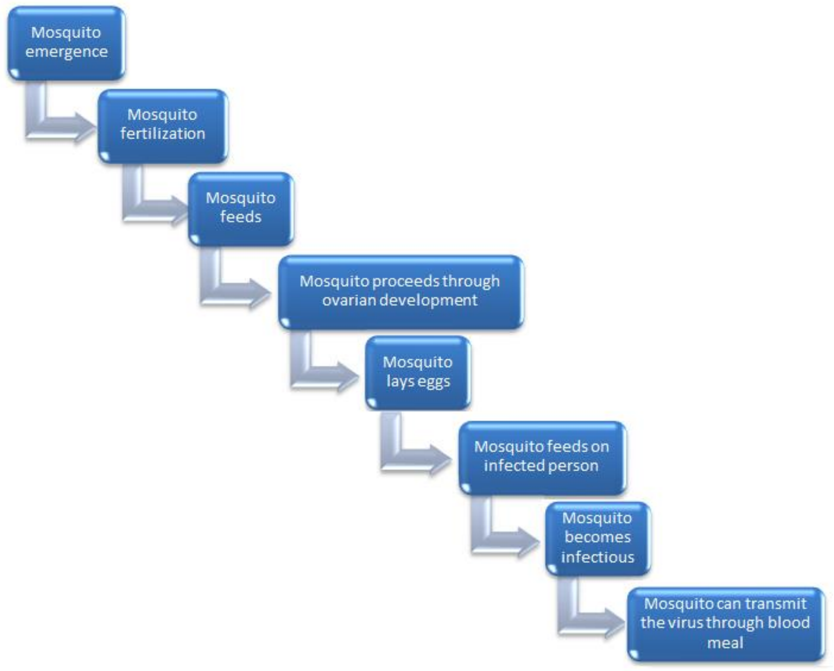
Mosquito emergence
Mosquito fertilization
Mosquito feeds
Mosquito proceeds through ovarian development
Mosquito lays eggs
Mosquito feeds on infected person
Mosquito becomes infectious
Mosquito can transmit the virus through blood meal

Figure 8: Adult life process

Table 8: Adult Survival

Temperature Range	Equation/Constant
T < 1, T > 40	.01
1 ≤ T ≤ 40	0.86

Ovarian development represents the rate at which the mosquito's eggs develop after taking a blood meal.

Table 9: Ovarian Development

Temperature Range	Equation/Constant
T ≤ 20	0
T > 20	(*.00898**((Temp + 273.15)/(298.15))*e^(*15725.23*/(1.987)*(1/(298.15)-1/(Temp + 273.15))))/(1+e^(*1756481.07*/(1.987)*(1/(*447.17*) -1/(Temp+273.15))))

The feeding rate represents the rate at which the mosquitoes are biting/taking blood meals. This only occurs if the ovarian development is not past 1, signaling that the mosquito is holding eggs. If it is already has developing eggs or is ready to lay them, then the mosquito will not feed.

Table 10: Feeding Rate (Bite Rate)

Temperature Range	**Equation/Constant**
T < 17, T > 39	0
17 ≤ T ≤ 39	-0.0066*T^2 + 0.3694*T - 4.2222

The Extrinsic Incubation Period (EIP) is the time required for dengue virus development in the mosquito. For other vector-borne diseases, it may be a parasite, such as *Plasmodium falciparum* that infects *Anopheles* mosquitoes responsible for malaria. In DyMSiM, this value needs to reach 1 for the mosquito to become infectious (Table 11).

Table 11: Extrinsic Incubation Period (EIP)

Temperature Range	**Equation/Constant**
T < 21	0
T ≥ 21	$(1/(100.33*e^{T}))/24$

The Vector Infection Rate calculates the probability of the adult mosquito becoming infectious after taking a blood meal from an infectious host, in this case humans. This means that even if a mosquito feeds on an infectious person, it still may not acquire the disease from that meal.

Table 12: Vector Infection Rate

Temperature Range	**Equation/Constant**
T < 17	0
T ≥ 17	-0.0023*T^2+0.1655*T-2.0568

RESULTS

Running the object-oriented model was done in the Eclipse Java integrated development environment (IDE). The model was run using daily weather data from the years of 2009, 2010 and 2011 in San Juan, Puerto Rico, for comparison against Stella. A container representing buckets or similar standing water areas in relation to a given household area was simulated. With these specifications, the model was run multiple times in order to view the amount of infectious adults per household over the two year period of study. Below is an example of the output given by the model:

```
Day: 314
Egg Batches: 166 Egg Population: 12152
Larvae Batches: 370 Larvae Population: 32228
Pupae Batches: 162 Pupae Population: 11949
Adults: 10421
Infectious Adults: 0
```

Figure 9: Sample output of the object-oriented model.

Model runs were able to be completed in about 50 minutes starting with a sole EggBatch of 65 eggs reaching peaks of 500,000 Adults.

Figure 10 is a comparison of the object-oriented DyMSiM model's outputs for 2010 and 2011 compared to the reported Dengue cases for those years. The model's output of infectious mosquitoes generally seems to fit the trend of the observed Dengue cases. There is a higher than expected amount of infectious mosquitoes for weeks 10 – 17 and 70 – 75, while a slight lower correlation than expected for weeks 22 – 25. The misalignment in these values, though, may be able to be explained due to the fact that the current model has a set Host Infection Rate, referring to the percent of the human population that is infectious. Holding this number constant, instead of calculating it based on human infections, prevents the model is from simulating the epidemic curve, which can be seen in the case data. The epidemic curve is generated as more humans become infected, due to a shortened incubation period in the mosquito. A higher infectious human population increases the probability of a mosquito taking an infectious blood meal. This generates more infected mosquitoes and creates a positive feedback loop resulting in an exponential increase in Dengue cases (see weeks 21 and 78, Figure 10). The epidemic levels begin to subside only after the mosquito population or the susceptible host population declines. The current object-oriented DyMSiM model does not link the mosquito infection rate to human infections, preventing the cycle from gaining momentum.

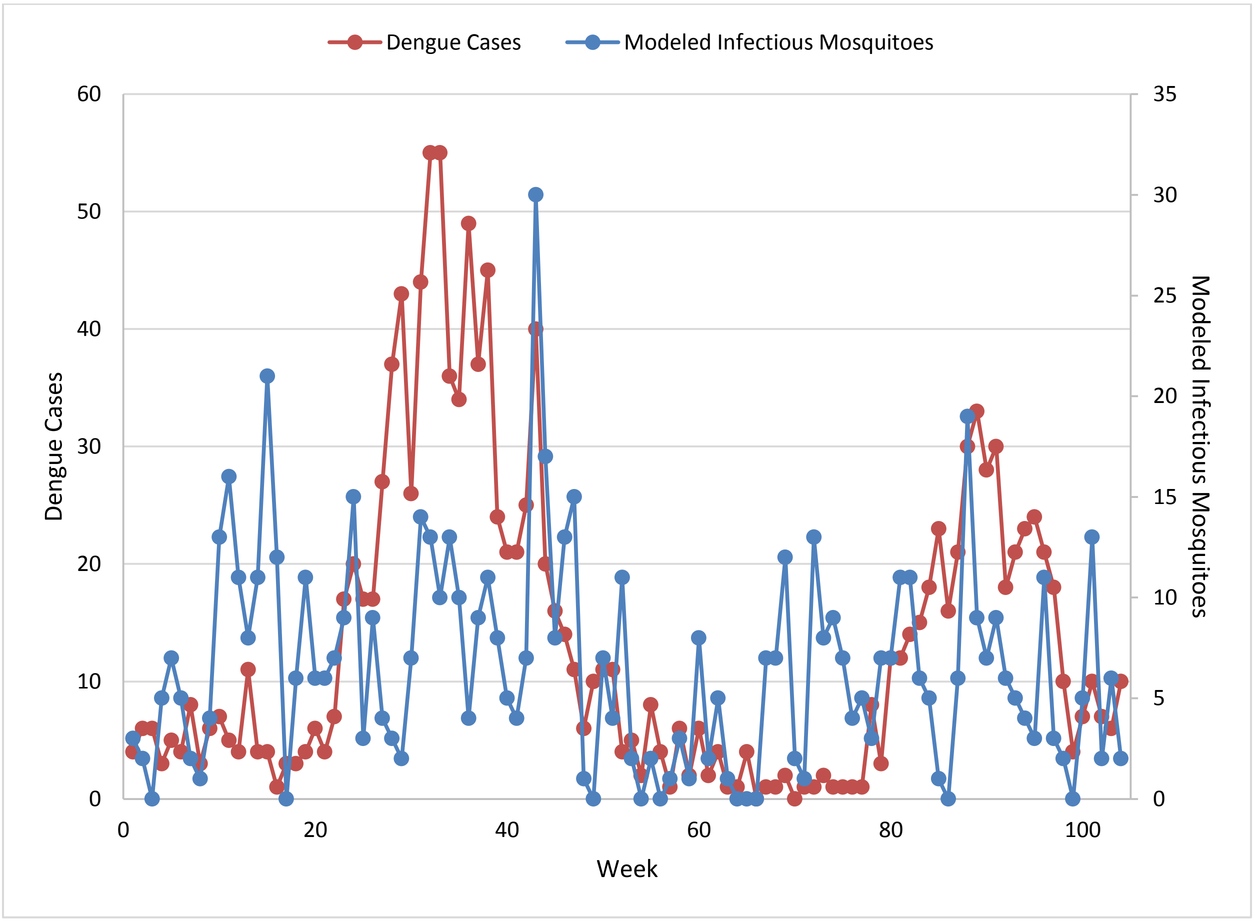

Figure 10: Modelled outputs of infectious mosquitoes compared against reported Dengue cases

CONCLUSIONS

The development of the object-oriented DyMSiM model produced a simple and efficient implementation in Java. Enhancements upon the Stella model included portability, usability and precision. While working on this project, multiple computers were used in the development process which was primarily enabled by the fact that it is a Java project. Any machine with Java installed was able to compile and run the model, a prospect that was simply was not available when running the Stella model. Usability was also enhanced, for example, with the usage of the Excel document inputting of the climate data. This functionality lends itself to many useful applications in the future development of this model, because of the already ubiquitous use of Excel, especially for logging climate data such as precipitation and temperature. Finally, the precision of the model was increased as well, especially in regards to the total amount of Adults in the Model. By being able to view the Adult objects as single entity, higher detailed study of the mosquitoes is allowed. When using the object-oriented model, a researcher can literally follow a mosquito's lifecycle from birth to death. The model, therefore, is able to give an extremely precise look at how the mosquito vectors behave in a spatial and temporal way while also being extremely accessible.

Due to the large amount of objects being tracked in the model, the performance of the model, unfortunately, was greatly decreased compared to Stella. The majority of the instantiated objects were that of Adults. This follows because for every one EggBatch, LarvaeBatch, or PupaeBatch will come at most 100 Adult objects (or whatever the specified EggBatch size is). Performance-wise it can be noted that the number of Adult objects is the determining factor in the memory and time consumption when running the model. In the example runs, it was seen that the model would take a severe plunge in its time consumption after Adult objects reached around 500,000 for a single Container. At this level, a daily run would take almost up to 5 seconds to compute, though years with smaller levels of Adults might take even less time than that. This obstacle, though, was overcome by simply using the model to view an ideal container that could act as a representative for a smaller spatial area. Potentially converting the Adult objects into a batch-type object similar to the other lifecycle stages may also be considered. Higher computing capacity access is also another solution to this problem if larger temporal or spatial areas are to be considered.

While this honors thesis was able to achieve many of the goals that were set out upon the beginning, there is still much left to be done. As discussed earlier, one of the main driving factors on creating the object-oriented DyMSiM was increasing its accessibility. There has been much excitement around the idea of creating a mobile version of DyMSiM or even simply making the model available online so it can work through a web browser. While the development of DyMSiM in Java readily enables the model to become available online, it is not ready yet to go to that phase. The most important effort that would need to be coordinated for a web-based DyMSiM would be the creation of a graphical user interface. Decisions on what variables should be modifiable by users must be made in order to design and create the best interface for the web-based version. Specifications dependent on mosquito species, geographic area, and disease component will all need to be set by a potential user. Moreover, making DyMSiM available on the web lends itself to the idea of DyMSiM as an educational tool. Features such as sliders for temperature and precipitation that show immediate results on mosquito populations would be an excellent device for students and educators to demonstrate climate dynamics, mosquito dynamics and the public health repercussions of climate change.

REFERENCES

Barbosa, P., T.M. Peters, N.C. Greenough. 1972. Overcrowding of Mosquito Populations: Response of Larval *Aedes Aegypti* to Stress. Environmental Entomology, 1: 89-93.

Christophers, S.R. *Aedes Aegypti*: The Yellow Fever Mosquito. The Syndics of the Cambridge University Press, 1960. London.

Forsythe, W.C., E.J. Rykiel, R.S. Stahl, H. Wu, R.M. Schoolfield. 1995. A Model Comparison for Daylength as a Function of Latitude and Day of Year. Ecological Modeling, 80: 87-95.

Kuno, G. 1995. Review of the Factors Modulating Dengue Transmission. Epidemiological Reviews, 17: 321-335.

Morin, C.W., A.C. Comrie. 2010. Modeled Response of the West Nile Virus Vector *Culex quinquefasciatus* to Changing Climate Using the Dynamic Mosquito Simulation Model. International Journal of Biometeorology, 54: 517-529.

Morin, C.W., A.C. Comrie. 2013. Regional and seasonal response of a West Nile virus vector to climate change. Proceedings of the National Academy of Sciences of the United States of America, 54: 517-529.

Morin, C.W., A.C. Comrie, K. Ernst. 2013. Climate and Dengue Transmission: Evidence and Implications. Environmental Health Perspectives, 11:1264-1272.

Muir, L.E., B.H. Kay. 1998. *Aedes Aegypti* Survival and Dispersal Estimated by Mark-Release-Recapture in Northern Australia. American Journal of Tropical Medicine and Hygiene, 58: 277-282.

"Vector-borne and Zoonotic Diseases." *Centers for Disease Control and Prevention*. Centers for Disease Control and Prevention, 14 Dec. 2009. Web. 2014.

"About Vector-borne Diseases." *WHO*. World Health Organization, 2014. Web. 2014.