Development of an Educational Platform for Teaching Young Students Complex Networking Topics using the Parallax BadgeWX

Lucas A. Yarnes and Corey E. Brady

Department of Teaching and Learning, Vanderbilt University, Nashville, TN, 37212

KEYWORDS. Modeling, Technology, Parallax, Learning

BRIEF. A platform was created that allows students to more thoroughly engage in group learning using models that teach complex networking concepts.

ABSTRACT. As the 21st century progresses, technology is integrated more and more fundamentally into the ways we think and learn about science, math, and our world. This drives a trend toward deeper and more conceptual uses. Not only does technology use make learning more engaging and memorable, but it is useful for expressing and exploring key concepts. Currently, there is limited technology that allows students to effectively explore these key concepts. Thus, the purpose of this project was to develop a platform that allows a wide range of students to acquire a deeper understanding of complex ecological and networking topics. The platform utilizes a relatively new piece of technology, the Parallax BadgeWX, to enable students to participate in engaging simulations to experience the emergent phenomena of the model firsthand. In this way, although each student focuses only on conducting his/her simple agent in the simulation, every student can see the large-scale pattern that emerges as a result of the interactions among individual students. In its final state, the platform is successfully able to be used to model a wide variety of topics that were previously unable to be effectively taught using models.

INTRODUCTION.

In the 21st century, technology is being integrated more and more fundamentally into the ways we think and learn about science, math, and the world around us. This drives a trend toward deeper and more conceptual uses of the technology. Not only does the use of technology make learning more engaging and memorable, but it is useful for expressing and exploring key concepts that arise in the classroom [1]. One implementation of technology in the classroom is through computational models, which are particularly useful for furthering student understanding of complex systems [2]. Some complex systems that can be found in nature, such as the spread of disease or the pollination of flowers, exhibit large-scale patterns that arise through the interactions of smaller agents (e.g. a bee during pollination), when the agents themselves do not exhibit that pattern. This is known as an emergent phenomenon. Agent-based modeling (ABM) allows students to break down these complex systems into simpler pieces and still observe the emergent behavior of the system.

One way that ABM can be introduced in the classroom is through a participatory simulation, whereby each student acts out an agent in the simulation. In this way, individual students are working together to follow a set of predefined rules, and as a result are able to observe the emergent phenomenon on the macro scale. This form of group role-play has been very effective in giving groups of students a sense of the emergent phenomenon, as it gives students a simple enough access point into a complicated system so that they can begin to understand all of its individual parts; however, this type of simulation has a few limitations that diminish its effectiveness in practice. Firstly, over the course of a simulation, sometimes mistakes are made that inhibit the emergent phenomenon from surfacing. Additionally, this type of simulation tends to be more individualistic rather than cooperative, which defeats its purpose [3].

Another way that ABM can be included in the classroom is through a purely virtual simulation. This type of simulation allows each student with a laptop to join a virtual classroom and control an agent in the simulation. Once again, though each student is focusing only on controlling his/her own agent, a classroom of students all controlling an agent can cause the emergent phenomenon to surface, though this time in a virtual environment. Although this type of simulation is much more precise and seamless, it lacks the embodied feeling that the role-play-style simulation has. Our goal is to create a hybrid form of the participatory simulation that includes the benefits of each type of simulation but mitigates or eliminates their limitations.

Our solution is to use technology to aid the student in modeling and allow for the flow of information between the agents in a role-play style simulation. For this application, it was crucial that the technology be easily modified both in hardware and in software and that the technology be inexpensive; so, we chose to use the Parallax BadgeWX, which was designed to fit this purpose well [4]. The BadgeWX is a wallet-sized wearable computer that is equipped with infrared sensors and a Wi-Fi module, and so can communicate with other nearby badges and with the Internet. The project focuses on developing a platform that integrates the BadgeWX with the modeling software Netlogo in order to extend the capabilities of a traditional offline role-play style simulation by allowing the agents in the simulation to seamlessly exchange information between one another. The goal of the platform is to not only make participatory simulations more useful and applicable to a wider range of topics, but also to make these simulations more engaging for the students and to ultimately increase the retention of the information presented through the simulation.

MATERIALS AND METHODS.

Phase 1: Learning and Familiarization.

The first goal of the project was to create a program that simulated the spread of disease using the badge's infrared (IR) sensors and its Wi-Fi module. The program was initially designed using pseudocode, which was then implemented using the badge's native language, C. On startup, one badge would become the first "infected" participant. Then, as the badge came into close contact with other badges in the simulation, the badge would "infect" the nearby badges using a command sent over the badge's IR sensors. Once a badge became infected, it would send a data packet to a web server dedicated to servicing the badges. On a computer, a NetLogo program would detect this new data packet and parse out the relevant information, which included the device identifier of the newly infected badge, the device identifier of the badge that infected it, and the time of the interaction in milliseconds (ms) since the start of the simulation. This process would continue until either all of the badges became infected or the user forcibly ended the simulation.

Phase 2: Development of the Wireless Communications Protocol.

Following the completion of the learning phase of the project, focus

shifted into working on the final product of the project. In order to do that, though, it was necessary to design a wireless communications protocol that utilized all of the functionality of the badge web server. The badge web API, as it is called, allows the badges to connect to a room that has a unique identifier, then post different kinds of data to their individual space in the room. When a badge joins, it receives a unique bucket identifier that it can use to post and receive data. When posting data, the badges can either send a data point or points to their bucket and it will get added to an ongoing list of data or they can post the data to a signal channel that only holds the latest value for the signal. The wireless protocol that was designed included the ability to perform all of these functions, both through NetLogo and through the badge. While designing the protocol, it was imperative that requests made to the server were made as infrequently as possible, so as to ensure that the server did not become overloaded and to ensure that the badge's program did not block too often. Additionally, it was important to ensure that any data sent to or from the badges or NetLogo arrived at its final destination, and that no data was lost or overwritten accidentally. To do this, the protocol was designed so that before any data could be written to the signal channel of a participant, the previous signal must be acknowledged somehow.

Once finished designing the protocol, a framework was implemented in both a C and a NetLogo program for testing. First, NetLogo was used to test the capabilities of the badge web API, since NetLogo is much less prone to bugs than the badges. One NetLogo program was written to be the master program, and another to simulate the badge. The master program was able to send and receive data to and from any badges that connected to the virtual room. If the master program needed to send information to a single badge, it would write the data to the badge's signal channel. If the master program needed to send data to all of the connected badges at once, it would write the data to its own signal channel, and the badges would continuously check to see of the data changed. Otherwise, every few seconds, the master program would check its data stream for new data points, and if it detected new data, would read and interpret them. The simulated badge program was only able to send data to NetLogo's data stream channel or read its own signal channel, which the master program was able to modify. Additionally, the simulated badge program would need to occasionally check to see if the master's signal channel changed.

With the wireless communication framework constructed in the two NetLogo programs, a C program was written that would be loaded onto a badge. The simulated badge NetLogo program was translated into an equivalent C program that the badge could run, to test if the badge was indeed capable of reading and writing data from and to the badge web API. After lots of debugging and trial and error, the C program was finished, and the badge could communicate with the master NetLogo program via the badge web API.

Phase 3: Development of the Final Platform.

With preliminary badge testing complete and the web protocol designed and implemented in both a C program and a NetLogo program, it came time to create the final platform. Because it is a platform and not a specific use-case program, the platform needed to be very general, with the end goal of creating a network of interactions and not simulating a specific scenario. Then, with the network of interactions saved in an external file, it could be imported into any number of other programs that could use the data to create an actual computer simulation of some phenomenon. In order to generate a network of interactions, there needed to be a motive for the users of the badges to interact with one another. For the purposes of this project, a simple game was created to do just that. Due to the unreliability of the badges, the game was hosted through the master NetLogo program, and the badges sent data directly to NetLogo. Once all of the badges had connected to the virtual room and NetLogo recognized that they were all there, the user

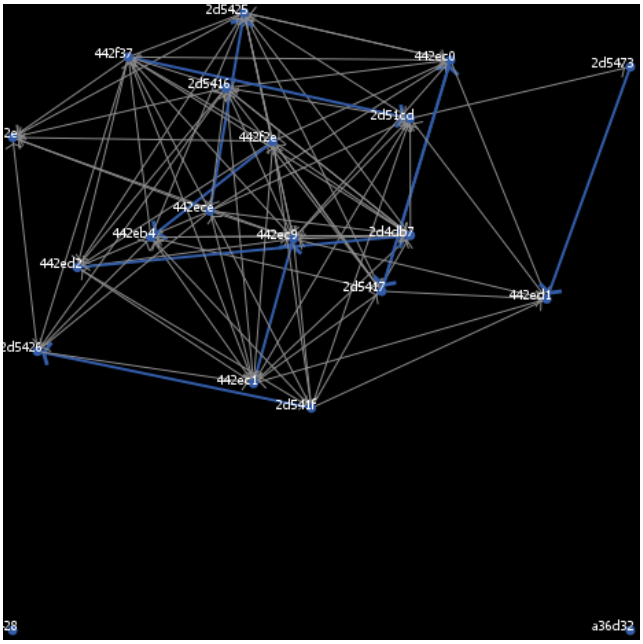

Figure 1. The view generated by NetLogo after a run of the simulation with 20 badges connected. Each badge is represented by a blue dot accompanied by a label that displays the badge's device ID. The grey lines between dots represent interactions between badges, and the blue lines represent interactions where the badges had the same code.

could start the game. On starting the game, each badge would be assigned a code that only one other connected badge had. As the users walked around the room, they would hold their badge to face another badge and press a button. The button press would cause one of the badges to send its unique device identifier to the other badge via the IR sensors. Once the other badge received the device identifier, it would send its device identifier and the identifier it received to the web server. When NetLogo received the pair of identifiers, it would compare the codes assigned to those identifiers to see if they were the same. If they were the same, NetLogo would send a command to the two badges that would cause the badges to flash a blue LED, alerting the users that they found their match. Otherwise, if the two badges did not have the same code, the LED's would flash green, alerting the user that NetLogo received the interaction, but that it was not a match.

RESULTS.

Using this simple game, a room of participants wearing the badges would be able to generate a network of interactions that could be saved to an external file for later use. Additionally, the master NetLogo program included a feature to see the results of interactions in real time. Figure 1 shows an example simulation of 20 participants that played this game.

The dots in the view represent badges (participants) and the lines between the dots represent the interactions. A grey line means the badges interacted but their codes did not match. If the line is blue, however, the badges interacted and their codes were the same. Once a simulation was complete, the user had the option to replay the simulation in the view, like a movie. The user could see the interactions appear in the view over time, exactly like how it was during the real simulation. Then, the user could export the interaction data to a file that could be used later to simulate specific scenarios, such as the spread of disease (Figure 2).

In this simulation, the user must first load a file containing data from a previous simulation into NetLogo. Then, the NetLogo program parses the data file into a useful format, determines how many badges

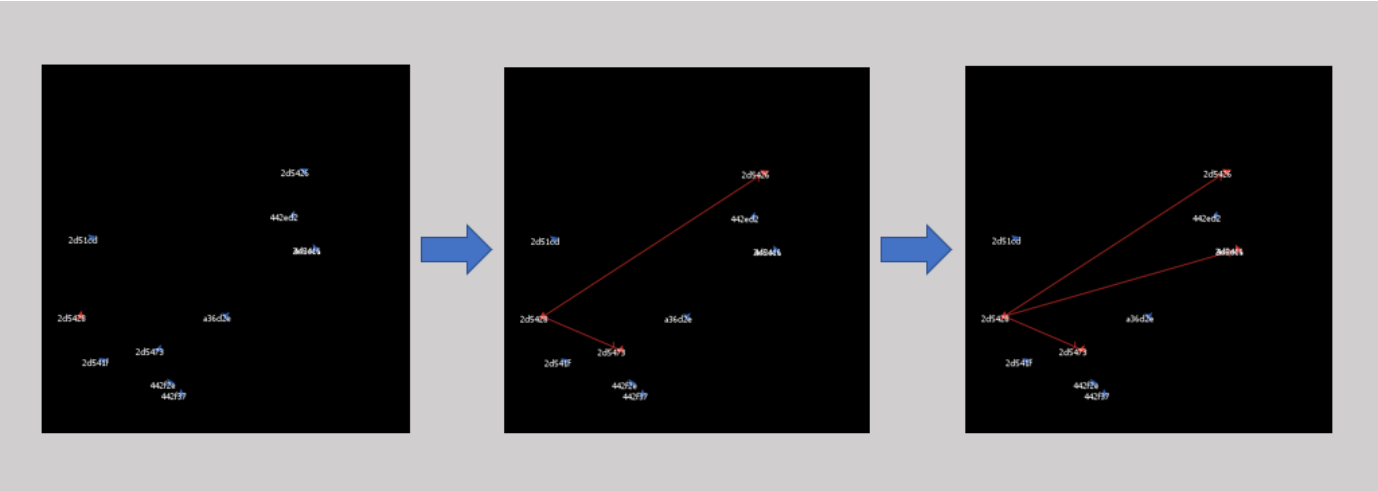

Figure 2. The view of a disease spreading simulation over time. Each dot represents a participant from a set of interaction data and is accompanied by a label displaying the device ID of the badge. A red dot means that the participant is infected. A red line connecting two dots shows the disease spreading to the next participant.

there were in the simulation, and creates a dot for each user. Then, one badge is randomly selected to be the first infected, and the simulation begins. Using the time information from the data file, NetLogo will play out the simulation movie-style, over time. As the simulation progresses, NetLogo checks to see if the interaction occurred between an infected badge and a non-infected badge. If this is the case, a line forms between the badges in the view and the non-infected badge becomes red, indicating that it is infected.

DISCUSSION.

This badge-based platform can take the participatory-simulations approach to understanding complex systems and emergent phenomena to a new level. Whereas participatory simulations tend to focus on students role-playing the behavior of agents in a system, if they incorporate badges, learners' interactions can exchange information and alter the content of future interactions. In this way, the badges can become co-participants in making the emergent phenomena appear. Focusing on and computationally manipulating information flow may also allow for the designing of participatory simulations that model new types of phenomena, expanding the range of topics that groups of students can investigate in this way.

As mentioned briefly before, one major limitation of this project was the badge itself. The badge is a new piece of technology only recently released to the public by Parallax Inc., and as such still contains many software and hardware bugs. Although the bugs did make development a more laborious task, they did not altogether inhibit progress. One area where the badges could be especially effective is in a scenario simulating the spread of disease. In this simulation, each participant could wear a badge and communicate verbally with other users, as they normally would. Meanwhile, the badges would track all interactions and send that information to a waiting Netlogo program. Then, after the simulation ends, the interaction data could be played back in Netlogo, and disease could be introduced in the view. Although the badges only tracked badge- rather than physical-interactions, the Netlogo program could show how a disease introduced at one point in the network would spread to the other participants over time. Additionally, a Netlogo program could introduce variables such as chance that the infection spreads, chance that the person being interacted with is immune to the disease, and a number of starting points for the disease. Netlogo could also track the spread of several diseases throughout the network at the same time, and each disease could behave in a different way. In short, although the badges are only sending information regarding interactions to the server, the data itself could be applied in a variety of different ways to serve the individual needs of the teacher using the technology in a classroom.

This platform has been demonstrated to a class of 20 graduate students on Vanderbilt's campus. That was helpful because the students were able to provide feedback such as ease of use of the platform, how consistent the badges were, and if the platform is viable for teaching emergent phenomena in the first place. However, in the future it would be more beneficial to test the platform on a group of younger students. Since the platform is designed to be used with middle- and high-school aged students, testing with a population closer to that age would provide more insightful feedback into the effectiveness of the platform for its original purpose: level of engagement and retention of knowledge.

An additional next step is to create other NetLogo programs that would use the interaction data to simulate different scenarios, such as pollination of flowers, natural selection of traits over generations of a species, and language migration or evolution over time. Although the platform itself could be used as a teaching tool if the game played by the users was changed, it would be just as beneficial to show some of the implications of the interaction data through external simulations. The disease-spreading program has already been created, but that is just one scenario out of many that could be utilized.

ACKNOWLEDGMENTS.

I would primarily like to thank Dr. Corey Brady for his assistance throughout the duration of the project, both in brainstorming the directions of the project as well as general help when problems arose. I would also like to thank my mentor, Dr. Elizabeth Deweese, for her help in revising my manuscript and the School for Science and Math at Vanderbilt (SSMV) for the opportunity to do this research.

REFERENCES

1. C. Brady, K. Orton, D. Weintrop, G. Anton, S. Rodriguez, U. Wilensky, All Roads Lead to Computing: Making, Participatory Simulations, and Social Computing as Pathways to Computer Science. *IEEE Trans. Educ.* **60**, 59–66 (2017).
2. C. Brady, D. Weintrop, G. Anton, U. Wilensky, Constructionist Learning at the Group Level with Programmable Badges, 8 (2016).

3. M. Resnick, U. Wilensky, Diving into complexity: Developing probabilistic decentralized thinking through role-playing activities, *The Journal of the Learning Sciences* **7**, 153-172 (1998).
4. C. Brady, D. Weintrop, K. Gracey, G. Anton, U. Wilensky, The CCL-Parallax Programmable Badge: Learning with Low-Cost, Communicative Wearable Computers, *Proceedings of the 16th Annual Conference on Information Technology Education - SIGITE '15*, 139–144 (2015).

Lucas Yarnes is a student at Martin Luther King Jr. Academic Magnet High School in Nashville, TN; he participated in the School for Science and Math at Vanderbilt University.