Week-4-Wendy-Huang-Daniel-Wendel-540p

WENDY HUANG: Hi, I'm Wendy Huang.

DANIEL WENDEL: And I'm Daniel Wendel. We're researchers from the MIT Scheller Teacher Education program.

WENDY HUANG: This week, we will be talking about constructionism and its connections to modeling and computational thinking.

DANIEL WENDEL: This week's lesson has four parts.

WENDY HUANG: In the first part, we'll talk about constructionism. By the end of the lecture, you will be able to define constructionism, explain how it's applied in education, and understand how it's related to modeling and computational thinking.

DANIEL WENDEL: Next, we will demonstrate how to program an epidemic model using the Toolblox agent-based modeling program.

WENDY HUANG: Finally, we will summarize the key points and then explain this week's assignment. What is constructionism? Seymour Papert was a MIT professor from the 1960s to the 1980s, and he invented a new theory of knowledge building called constructionism. Papert defined constructionism as "building relationships between old and new knowledge and interactions with others, while creating artifacts of social relevance."

Papert built upon Piaget's theory of constructivism, which gave a central role to the learner and his or her active engagement in knowledge construction. But whereas Piaget saw formal abstraction as the goal of knowledge construction, Papert valued both the abstract and concrete forms of knowledge and the social contexts where such learning occurs. When thinking about education, a constructionist focuses on how learning is constructed and the interactions between teachers and students as they engage in the design and discussion of learning artifacts.

Papert is probably best known for inventing LOGO, a programming language that featured a turtle that can follow simple movement commands to draw on a computer screen. LOGO allowed children to (1), identify with a computational object; (2), manipulate the turtle in ways similar to objects in the real world; and (3), connect personal experiences to mathematical concepts and operations. Computer models are a type of object to think with.

Using models allows learners to manipulate mathematical and scientific objects and conduct inquiry in new ways. Modifying models and creating models are even more constructionist in nature, because they engage students in a design process to produce objects to think with that students can share and discuss with other people. This week, we will focus on the modify and create parts of the Use, Modify, Create progression.

Here are some questions that have guided our design our agent-based models, like the EvoFish model you used last week, as well as other examples in the video segment about complex systems. We will use these questions to design and construct an epidemic model for this week's assignment.

The questions are" What is the system or process you want to model? What are the features of interest to model? How will the model visualize these features? Who or what are the agents? How do the agents behave and interact with each other or the environment? And what tool will you use to build and test the model?

DANIEL WENDEL: In our example epidemic model, we have answered the questions like this. What is the system or process we want to model?

WENDY HUANG: An epidemic, which is to spread of a disease through a population.

DANIEL WENDEL: What are the features of interest to the model?

WENDY HUANG: We hypothesize that parameters, such as starting numbers of sick and healthy agents, as well as population density will affect the rate of the disease spreading. We also want to track the numbers of healthy and sick agents over time.

DANIEL WENDEL: How will the model visualize these features?

WENDY HUANG: Using colors.

DANIEL WENDEL: Who or what are the agents?

WENDY HUANG: Agents are the individuals that make up the population. They can be healthy or sick.

DANIEL WENDEL: How do the agents behave and interact with each other or the environment?

WENDY HUANG: Agents will move around randomly. Sick agents spread disease through contact.

DANIEL WENDEL: What tool will we use to build and test the model?

WENDY HUANG: Toolblox, our online agent-based modeling program.

DANIEL WENDEL: Computational thinking is used in the build step of the design process. Computational thinking involves breaking down the behaviors of the agents into precise rules that a computer can understand and follow. This process is called programming. Breaking down larger ideas into several simpler ones in this way is central to computational thinking, but it's also useful whenever a problem is too large to solve all at once.

Modeling and computational thinking are iterative processes, which means that you go back to the beginning many times. This is because as you test the model or program you are building, things you learn often cause you to change your original design. For this reason, it is best to make small changes and test often as you are building, rather than waiting until you have finished only to realize that your program or model does not work the way you hoped it would. Or even although it does work the way you hoped it would, your original thinking was wrong, so you need to go back and make large changes.

Again, test often, and don't be afraid to change your plans if you need to. For this week's assignment, you will use Toolblox to program a model that stimulates the spread of a disease through a population. You will also design and program at least one model extension. We will now demonstrate how to program a simple epidemic model using Toolblox. Don't worry if you don't understand or remember everything.

We will provide you with a step by step tutorial to build the epidemic model. This is Toolblox. Before you can build a model using Toolblox, you need to create an account. To create an account, click on the Register for Account link. Choose a username and password, and enter your email address if you want us to be able to send you an email if you forget your password.

Once you have created your account, click on the My Profile link. Then click the Create a New Project button to create a new project. We will be starting from a blank project, so choose that, and then click Submit.

This is the Toolblox programming environment. At the top of the page is some information about the project, such as a title, description, and author. Change the title and description to be something useful. Below the project information is the 3D world where the model will run. We call this area Web Land.

You have already seen this area during the last week's assignment. The buttons in this area are used to control the model. As a reminder, you can move the view by dragging and scrolling with your mouse. This is the programming workspace, where you will program the behaviors of your agents.

Notice the drawer is full of command blocks on the left. You create your program by dragging blocks from these drawers and connecting them on the pages of the agents on the right. For example, if I want The World to create 300 turtle agents when the Setup button is pushed, I open the Interface drawer and drag the When Button Pushed block onto The World page.

I choose Setup from the drop down. Then I open the Agents drawer and connect a Create Do block inside that first block, and change it to 300 turtles. I chose the Create and Do block, because I also want to make the turtle scatter, which is another block in the Agents drawer. I also want them to turn yellow when they are created.

To turn them yellow, I open the Traits drawer. And I use the Set block and Color block. I choose Color from the first drop down and yellow in the second one.

Now, that I've done this, it's time to test. I scroll to Web Land and click the Setup button. And sure enough, 300 yellow turtles appear. My turtles look like cubes, but you can choose a different shape if you want by following the instructions on the worksheet. Now, I'll click Setup again.

Oh no, now there are 600 turtles. In order to make the number reset every time, I have to add one more block. From the Agents drawer, I choose the Delete Everyone block. Make sure to put this block before the Create block, otherwise it would delete the new agents you just created.

Now, if I go back and click on Setup, the old turtles are deleted, and I only have my 300 new turtles. Next, I need to change the program so that some turtles start out infected, which we represent by turning the turtles red. If I go to the Traits drawer and add the blocks to set my color to red, this will make all of the turtles turn red.

To make only some of them red, we need to open the Logic drawer and get an If block. We want 10% of the agents to start out red. To do this, we can open the Math drawer and use the Random block. This block gives a random number between one and the number you put.

If I put the number 100, it will give me a random number between one and 100. Since the number is random, about 10% of the time, it will be a number less than or equal to 10. So I can open the Logic drawer and get the Less Than Or Equal To block.

I can write in a 10 and connect the blocks together. Now, this code will only run approximately 10% of the time. So if I put the Set Color to Red block inside that If block and connect these blocks together, now I see 300 turtles with about 10% of them being red and the rest being yellow. Now, it's time to program the behavior of the turtle agents.

So I will switch to the Turtle page. All of the turtles will follow whatever commands I program on this page. For this simple model, we just want the turtles to wander around while the model is running. To do that, we need some Movement blocks, like Forward, Left, and Right.

We also need, from the Interface drawer, a While Button Toggled block. And choose Forever from the drop down. I can connect the Forward, Left and Right blocks here. But if I want the turtles to wander, rather than walking in a pre-determined pattern, I also need the Random block from the Math drawer.

If I make the turtles turn a random number of degrees left and right, they will follow an unpredictable wandering path. Now that this is programmed, let's test by clicking the Forever button in Web Land. Sure enough, the turtles seem to be wandering around, moving forward but also following an unpredictable path. However, notice that nothing happens yet when red and yellow turtles collide with each other.

To program this aspect of the model, we will open the Detection drawer and use an On Collision With block and choose Turtle from the drop down. When a turtle collides with another turtle, we need it to check to see if the other turtle is red. If so, the total will turn itself red as well. To do this, we need an If block, also, an Equals block and from the Traits drawer, a Trait Of block and a Color block.

Finally, from the Detection drawer, we need the Collidee block, which tells us who the turtle is that we just collided with. Now, the code says on a collision with another turtle, if that other turtle's color equals red, then we need to do something, which is again, from the Traits drawer, set my color to the color red.

That's it. Now, if we test the code by clicking Forever again, you should see the yellow turtles turning themselves red whenever they collide with red turtles. And they do. It looks like the model is working so far. There are a few more things to do in order to have a more complete epidemic model, but we will let you explore those in your own model.

WENDY HUANG: As we built our simple epidemic model, some questions came to mind that caused us to rethink our design. What if the agents can recover, that is, become healthy again after being sick? What if some agents are immune and can't get infected?

What if agents recover becoming immune? How would these new behaviors affect the model? What we're doing what we think about and answer these questions is iterating back to the design process. We are adding some new behaviors to make the model better reflect the real world.

Next, we will iterate within the build process by using computational thinking to come up with precise rules to describe the behaviors. For example, if agents can recover, how should they recover? Should they just recover after a certain period of time? If not, how do we define recovery rules for a population where different agents take different amounts of time to recover?

This is one area you will be able to investigate in your own model. To summarize, modeling involves going back and forth between design, build, and test, and also iterating within each step of the process. Iterate as much as you can. In summary, to key points of our lesson are constructionism is a theory of knowledge building that emphasizes the role of constructing artifacts in a social context.

DANIEL WENDEL: Modeling is one such expression of constructionism.

WENDY HUANG: Using a model design process involves making one's mental model more explicit.

DANIEL WENDEL: Then we can use computational thinking to construct the model as a computer artifact.

WENDY HUANG: This week, we're focusing on the Modify and Create aspects of the Use, Modify, Create progression.

DANIEL WENDEL: And we've shown you how to program a simple model in Toolblox.

WENDY HUANG: For this week's assignment, you will read an article titled "Constructionism," which goes into greater depth about this theory and its applications. In this week's online discussion forum, please answer the question, "How would you convince a teacher that constructionism is important for teaching and learning?" What examples would you give?

You will also program an epidemic model in Toolblox by following the directions on the epidemic programming tutorial. Then you will choose or invent one extension to program. It needed, you may ask for programming help via an online discussion forum or during the scheduled live chat. Finally, you will write a reflection about the activity and provide a link to a Toolblox epidemic model in the online discussion forum.

DANIEL WENDEL: Thanks for watching, and we look forward to seeing your responses in the course website.

WENDY HUANG: Here's our contact info one more time.