Paper Title Characterizing the Nature of Student Theory Building in the Context of Computational Modeling Activities

Author(s) Hillary Lucille Swanson, Utah State University; Kit Martin, Northwestern University; Bonni Jones, Utah State University; Siyu Wu, Utah State University; Maryam Hedayati; Bruce Sherin, Northwestern University; Uri J. Wilensky, Northwestern University

Session Title Science Education in Computer-Supported Collaborative Settings

Session Type Roundtable Presentation

Presentation Date 4/9/2021

Presentation Location Virtual

Descriptors Computers and Learning, Science Education

Methodology Qualitative

Unit SIG-Learning Sciences

DOI https://doi.org/10.3102/1691190

Characterizing the nature of student theory building in the context of computational modeling activities

Abstract

It is widely agreed that engaging students in authentic science practices is important for science education. Theory building is a central practice of science. Today, many scientists build theory through computational modeling. This paper presents a block-based computational modeling activity to support students' engagement in building theory about the spread of disease. We characterize the work of one student, Sage, in the context of her construction of models of Ebola, the flu, and a zombie apocalypse. Using grounded analysis, we identified 37 moves involved in Sage's theory building, related to her refinement of models, as well as meta-knowledge about the nature of the models. We present these moves and illustrate them using data from Sage's construction of the three models.

It is widely agreed that engaging students in authentic science practices is important for science education (Duschl, Schweingruber, & Shouse, 2007). Theory building is a central practice of science. Today, many scientists build theory by constructing computational models that, when run, produce outcomes that can be explored and compared with experimental findings (Weintrop et al., 2016; Foster, 2006).

A number of research programs have explored ways of engaging students in theory building through computational modeling. There is a long tradition of asking students to create models of phenomena from Newtonian physics. diSessa (1995) describes a case where high school students re-invented F=ma through their development of computational models. Sherin (2001) looked broadly at the possibility of using programming as a language for expressing simple physical ideas. Wilensky and colleagues have investigated student engagement in computational modeling of complex systems phenomena, such as predator-prey dynamics, using NetLogo (Wilensky, 1999; 2001; Wilensky & Reisman, 2006). Recent work has examined student construction of models using NetTango (Horn & Wilensky, 2011; 2012), a block-based interface to NetLogo. These studies have examined students' development of both scientific understanding and computational thinking through their construction of models (Horn et al., 2014; Wagh & Wilensky, 2017).

The present work builds on this tradition by examining the nature of student theory building in the context of computational modeling activities. It seeks to characterize elements of theory building enabled and supported by block-based microworlds.

Theoretical Foundations

We define scientific theory building as a family of practices through which scientists systematically refine theoretical knowledge artifacts, including laws, models, explanations, constructs, and categories (Author, 2020a). As artifacts are refined, thinking is refined. Our perspective aligns with Einstein's (1936) notion that "the whole of science is nothing more than a refinement of everyday thinking," and constructivist frameworks that view the construction of new knowledge as a refinement of prior knowledge (Piaget, 1971). It also aligns with constructionism (Papert, 1980), which argues that learning happens best through the construction of public artifacts, such as computational models. In our work, we seek to characterize students' theory building by describing the moves through which they refine their computational models. In this paper, we focus specifically on characterizing the process of student theory building, leaving the science learning that results to other papers.

Methods

Study Goals

This paper presents the results of an analysis of data taken from a larger study focused on scaffolding student engagement in different approaches to scientific theory building, including the construction of agent-based computational models. We are iteratively refining block-based microworlds using the NetTango interface to NetLogo. NetTango makes the computational power of NetLogo accessible to authors by using a block-based programming language curated to a particular phenomenon. NetTango blocks are not a full programming language, but domain-specific blocks relevant to the modeled phenomenon. Previously called *semantic blocks* *(*Wilkerson-Jerde & Wilensky, 2010) and now called *domain blocks* (Wagh et al., 2017) the blocks are primitive elements of code that represent agents' actions, which can be combined to model a specific phenomenon. We are designing domain-block libraries for simulating complex systems phenomena and studying how children use the blocks to engage in scientific theory building. This study asks the question *"what is the character of student theory building in the context of computational modeling?"*

Study Design

To address this question, we tested NetTango models with middle school students (ages 12-14) during one-on-one 1.5-hour task-based interviews. During each interview, the student had full command of a laptop with an agent-based microworld. The interviewer guided them through tasks and questions from a semi-structured protocol, which introduced the features of the microworld and then prompted the student to model a particular phenomenon (e.g., an epidemic of a disease of their choice).

This study focuses on an interview with one student, Sage. Sage was 13 years old and had just started 8^{th} grade at a public school in her small Midwestern city. Sage explored the *Spread of Disease* model, shown in Figure 1. The screenshot to the left shows the agent-based microworld before a model has been built. The screenshot to the right shows the microworld with a model that has been built and initialized. In both screenshots, the black box to the left is the *world* which depicts the activity of the agents that are programmed to behave according to the rules specified by the model. The *setup* and *go* buttons are controlled by procedures (red blocks) that the user must drag from the block library (far right) into the modeling field (middle) and then define by connecting with blocks (purple, grey, and green), such as *move*, *if contact person*, and *infect*.

[Figure 1 goes here]

Sage's interview was recorded using video, audio, and screencast technology. The audio recording was transcribed. A fine-grained grounded analysis was applied to both the screencast and interview transcript to identify theory-building moves that Sage enacted. First, the screencast of Sage's entire interview was reviewed and times were noted during which she engaged in building models for particular diseases, namely Ebola, the flu, and a zombie apocalypse. These episodes were then marked on the transcript, which was then read for evidence of theory-building moves that corresponded with basic categories determined in prior research (Authors, 2020b). These categories were 1) initial articulation moves, 2) testing moves, 3) refining moves, 4) applying moves, and 5) modeling meta-knowledge. The moves identified in the transcript were coordinated with screencast recordings to get a sense for the student's actions in the microworld and develop a more complete picture of her theory-building moves.

Findings

The grounded analysis revealed 37 theory-building moves across the five categories. The general categories and specific moves are outlined in Table 1 and then introduced (in italics) and briefly unpacked below. They are described in greater detail and exemplified in Tables 2-6, in Appendix B.

[Table 1 goes here]

Initial Articulation Moves

Sage crafted her initial model through *initial articulation moves,* including *recounting prior knowledge*, *initial planning*, *determining relevant code*, *specifying agent rules*, *purposefully selecting and approximating parameter values*, *deciding how to model time*, and *reaching for and making sense of available resources*. For example, in her initial construction of a model of Ebola, Sage began by describing what she knew about the disease and how this might be represented in her model. She then looked through the available code and determined that blocks like "infect in a radius" were less useful to her model, because her understanding of Ebola was that it was transmitted through direct contact. She dragged code-blocks into the authoring space to create a basic model where people would infect each other with some probability when they made contact. She asked if she could search for information about the disease online, and translated what she found into approximate values for parameters including probability of infection, death and recovery.

Testing Moves

Sage tested her model through *testing moves*, including *predicting and explaining the outcome of a model run*, *planning for purposeful exploration*, *testing parameter settings or agent-rules*, *comparing trials*, *slowing down a model run*, *observing model behavior, noticing how a model implements code*, *comparing results of a model run with predictions*, *evaluating a model-run outcome and explaining its cause*, and *comparing the modeled phenomenon with other phenomena*. For example, in her construction of a model of the flu, Sage predicted the people would spread the infection much more quickly than Ebola, noting that the probability of dying was much lower in the case of the flu, so that people should live long enough to transmit the disease. When the rate of transmission was still not as high as she had expected, she announced that she wanted to collect a dataset and compare runs with different probabilities of infection, and that she wanted to slow the model to see what was happening in agent interactions when the disease died out.

Refining Moves

Sage modified or debugged her model through *refining moves*, including *noticing a problem and modifying code blocks*, *modifying parameter values for ease of mental mathematical computations*, *modifying code to simplify the model*, and *debugging thinking*. For example, before testing her initial model of Ebola, Sage noticed a problem with the code: a command for "move" was missing from the procedure. She noted that this wouldn't work - the people would stay where they were and no one would get anyone else sick. She added the "move" block to remedy the situation. She also modified the number of initially healthy and sick people so that they would add to 100, for ease of comparing later ratios with initial ones. She debugged her thinking and refined her model when her first model run produced a surprising result: within several ticks everyone in the world was healthy. She attributed this to the very high death rate and lowered the probability of death to get the disease to spread. After constructing and testing her model of the flu, she noted that surprisingly, the flu epidemic was more deadly at the population level, despite Ebola being the more deadly disease at the individual level. This is a notable shift in understanding, which shows that Sage's engagement in computational modeling is helping her to resolve a commonplace confusion regarding level of causality (Wilensky & Resnick, 1999).

Applying Moves

Sage used the model to make sense of phenomena through *applying moves*, including *describing the outcome of a model run*, *interpreting numerical readouts*, *coordinating data from multiple readouts in the interface*, *referencing data*, *making sense of outcomes*, *explaining the* *aggregate-level phenomenon as a result of agent-level interactions*, *comparing the model with the real world*, *comparing the modeled phenomenon with other phenomena*, *drawing conclusions about complex systems dynamics*, *assessing the reasonableness of results*, and *looking for key relationships*. For example, in her model of the flu, Sage interpreted the graph to draw conclusions about the rate the disease spread through the population. She coordinated between the graph and the readout of the number of people in the world to understand the role of the probability of death in the model. She also related the outcomes of her model to what she knew about the Spanish flu epidemic of the early 20th century.

Meta-Knowledge

Meta-knowledge did not consist of moves, but rather, elements of understanding Sage showed regarding the nature of her model and the activities in which she engaged, including *identifying limitations of the microworld*, *distinguishing critical from cosmetic components*, *noticing the approximate nature of the model*, *identifying how the approximate nature of the model may or may not impact model outcomes*, *awareness of the limits of her own knowledge*, and *reaching for credible resources*. For example, in her zombie apocalypse model, Sage wished she could program the zombies to move more slowly than humans in the world. She remarked she didn't think this would really make a difference, because she thought agent speed was "just cosmetic" and wouldn't actually influence the model's outcomes. In her model of the flu, she noted that the maximum number of people who could initially occupy the world was 400, and that this was considerably smaller than the population of a city. Unlike agent speed, Sage regarded population size as a factor that could change the dynamics, and therefore outcome, of the model run.

Significance

This study characterizes student theory building in the context of computational agent-based modeling. Findings suggest that such theory building is a highly complex activity, consisting of a constellation of moves related to the initial articulation, testing, refinement, and application of the model, as well as meta-knowledge concerned with the nature of models and modeling. The work makes a contribution to the larger project of characterizing the nature of student engagement in different forms of scientific theory building (Author, 2020b). More specifically, our findings offer insight into the nature of student work at the intersection of two scientific practices emphasized by the NGSS: modeling and computational thinking. Our work is foundational for the development of learning objectives for science curricula and assessments that capture the richness of student engagement in scientific theory building.

References

Author (2020a)

Author (2020b)

diSessa, A. A. (1995). Designing Newton's laws: Patterns of social and representational feedback in a learning task. In R.-J. Beun, M. Baker, & M. Reiner (Eds.), *Dialogue and Interaction: Modeling Interaction in Intelligent Tutoring Systems.* Berlin: Springer-Verlag, 105-122.

Duschl, R. A., Schweingruber, H. A., & Shouse, A. W. (Eds.). (2007). *Taking science to school: Learning and teaching science in grades K-8* (Vol. 500). Washington, DC: National

Academies Press.

Einstein, A. (1936). Physics and reality. *Journal of the Franklin Institute*, *221*(3), 349-382.

Foster I (2006) 2020 computing: a two-way street to science's future. Nature 440(7083):419

Horn, M.S., Brady, C., Hjorth, A., Wagh, A., & Wilensky, U. (2014). Frog Pond: A code first learning environment on natural selection and evolution. Proceedings of IDC 2014.

Horn, M.S., & Wilensky, U. (2011). NetTango [Computer Software]. Evanston, IL: Center for Connected Learning and Computer Based Modeling, Northwestern University.

Horn, M.,S. & Wilensky, U. (2012). *NetTango: A mash-up of NetLogo and Tern*. In Moher, T. (chair) and Pinkard, N. (discussant), When systems collide: Challenges and opportunities in learning technology mashups. Symposium presented at the annual meeting of the American Education Research Association, Vancouver, British Columbia.

Papert, S. (1980). Mindstorms: Computers, children, and powerful ideas. *NY: Basic Books*.

Piaget, J., *Psychology and Epistemology: Towards a Theory of Knowledge* (New York: Grossman, 1971).

Sherin, Bruce L. "A Comparison of Programming Languages and Algebraic Notation as Expressive Languages for Physics." *International Journal of Computers for Mathematical Learning* 6, no. 1 (2001): 1–61.Wadsworth, B. J. (1996). *Piaget's theory of cognitive and affective development: Foundations of constructivism*. Longman Publishing.

Wagh, A., Cook‐Whitt, K., & Wilensky, U. (2017). Bridging inquiry‐based science and constructionism: Exploring the alignment between students tinkering with code of computational models and goals of inquiry. *Journal of Research in Science Teaching*, *54*(5), 615-641.

Wagh, A., & Wilensky, U. (2017). EvoBuild: A Quickstart Toolkit for Programming Agent-Based Models of Evolutionary Processes. Journal of Science Education and Technology, 1-16.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*(1), 127-147.

Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

Wilensky, U. (2001) *Modeling nature's emergent patterns with multi-agent languages*. Proceedings of EuroLogo 2001. Linz, Austria

Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep, or a firefly: Learning biology through constructing and testing computational theories—an embodied modeling approach. *Cognition and instruction*, *24*(2), 171-209.

Wilensky, U., & Resnick, M. (1999). Thinking in levels: A dynamic systems approach to making sense of the world. *Journal of Science Education and technology*, *8*(1), 3-19.

Wilkerson-Jerde, M., & Wilensky, U. (2010). *Restructuring Change, Interpreting Changes: The DeltaTick Modeling and Analysis Toolkit*. Paper presented at the Constructionism 2010 Conference, Paris.

Appendix A: Figures

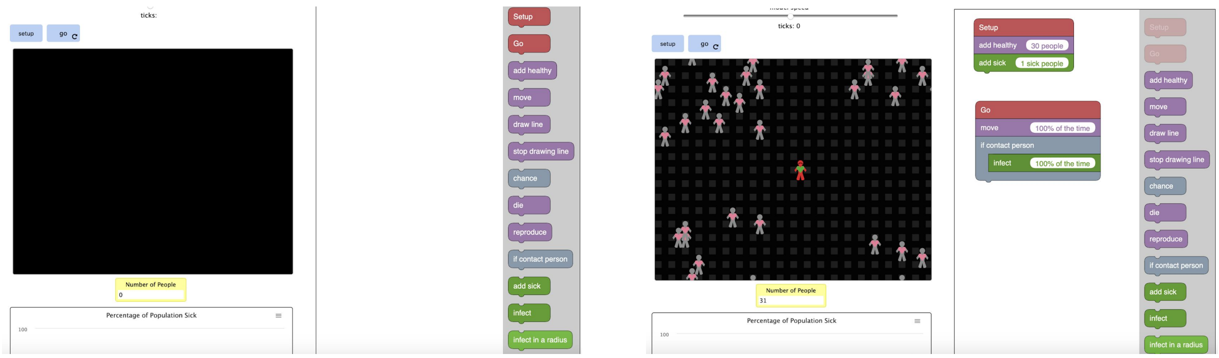

Figure 1. Screenshots of the Spread of Disease microworld before (left image) and after (right image) a model has been built.

Appendix B: Tables

Table 1. Theory Building Moves

Initial articulation moves	Testing moves	Refining moves	Applying moves	Meta-knowledge
Recounting prior knowledge Initial planning	Predicting the outcome of a model run Explaining the prediction	Noticing a problem Modifying code to solve a problem	Describing the outcome of a model run	Identifying limitations of the modeling environment

Determining relevant code Specifying agent rules Purposefully selecting and approximating parameter values Deciding how to model time Reaching for available resources Making sense of available resources	Planning for purposeful exploration Testing parameter settings or agent-rules Comparing across trials Slowing down a model run Observing model behavior Noticing how a model implements code	Modifying parameter values for ease of mental mathematical computations Modifying code to simplify the model Debugging thinking	Interpreting numerical readouts Coordinating data from multiple readouts in the interface Referencing data Making sense of outcomes Explaining the aggregate-level phenomenon as a result of agent-level interactions	Distinguishing critical from cosmetic components Noticing the approximate nature of the model Identifying how the approximate nature of the model may or may not impact model outcomes Awareness of the limits of one’s own knowledge

	Comparing results of a model run with predictions Evaluating a model-run outcome Explaining the cause of a model run Comparing the modeled phenomenon with other phenomena		Comparing the model with the real world Comparing the modeled phenomenon with other phenomena Drawing conclusions about complex systems dynamics Assessing the reasonableness of results Looking for key relationships	Reaching for credible resources

Table 2. Initial Articulation Moves

Move	Description	Example
Recounting prior knowledge	The student considers what they know about the phenomenon of interest.	The student recounts knowledge they have about getting sick and spreading disease, or stories they have heard about the particular disease.
Initial planning	The student considers what should go in the model based on their prior knowledge of the phenomenon.	The student knows that people spread germs by coughing and sneezing or through direct contact, and that how the germs are transmitted depends on the disease.
Determining relevant code	The student considers what should go in the model based on the available code blocks and their relevance to the phenomenon.	The student decides that "die" is relevant for a model of Ebola, but not for a model of the flu.

Specifying agent rules	The student specifies the rules of agent behavior and interaction, encoding their expectations by arranging blocks in the “go” procedure.	The student specifies that every tick of the clock, agents move, infect their neighbor if they are sick, and recover with some probability.
Purposefully selecting parameter values	The student purposefully selects the values of parameters based on their hypotheses of agent characteristics and initial conditions.	The student chooses a high probability of infection for a disease like the flu, because they think the flu is pretty infectious.
Approximating parameter values	The student approximates parameter values.	The student uses rounded values for the initial number of sick and healthy people in the world, for ease of calculation. They use “guesstimates” in the place of exact known values for parameters like probability of infection.

Deciding how to model time	The student considers how time should be modeled in the simulation.	The student might choose to have ticks represent minutes, hours, or days.
Reaching for available resources	The student reaches for available resources to supplement their prior knowledge.	The student looks through a textbook or conducts a web search.
Making sense of resources	The student makes sense of available resources, translating the information into ideas that are useful to their construction of the computational model.	The student looks at facts on the World Health Organization website for Ebola and uses calculated transmission rates to inform the value they select for the probability of infection parameter.
Describing expected model behavior	The student describes the behavior they expect the model to produce, based on the computational program they have built.	The student reads through the code line by line, checking it by describing aloud the sequence of actions the agents should enact.

Evaluating initial model code	The student evaluates the program they have written before running the model.	The student reads through the code and notes where they expect it to produce the expected outcomes, or where they are uncertain and perhaps experimenting.

Table 3. Testing Moves

Move	Description	Example
Predicting the outcome of a model run	The student describes the outcome they expect the model to produce, based on the parameter settings they have selected.	The student sets the probability of infection near 100% and predicts that everyone in the world will be sick after only a short number of ticks.
Explaining one's prediction for a model run	The student explains why they expect the model to produce the aggregate-level outcome they have predicted, by reasoning through the agent-level interactions.	The student explains that the graph of the percentage of infected people should follow an S-curve because at first very few people have the infection to pass on. When

		more people are sick, the rate of transmission will pick up with more people to pass the disease on. Finally, the rate of transmission will slow down with few people left who are susceptible.
Planning for purposeful exploration	The student makes a plan to run the model a number of times, varying parameters over particular values and noting their effects.	The student plans to explore how the rate of recovery influences the graph of the percentage of people who are sick over time.
Testing parameter settings or agent rules	The student tests a new parameter value or agent rules by running the model.	The student increases the number of people in the world who are sick at the beginning of the model run to see how that influences the model.
Comparing across trials	The student compares the outcomes of two or more simulation trials, modifying a	The student changes the infectiousness of the disease and compares the number of

	parameter from trial to trial.	clock-ticks it takes for everyone in the world to get sick.
Slowing down a model run	The student might slow down the speed of the model run in order to see how the programmed interactions play out in the simulation.	The student lowers the model-run speed in order to observe the individual agent interactions that result in a model where everyone dies within a few ticks of the clock.
Observing model behavior	The student describes what they see happening in the model as it runs.	The student announces that the number of people infected with Ebola is increasing.
Noticing how a model implements code	The student makes sense of how the model implements the code by experimenting with the blocks in their program.	The student adjusts the infectiousness parameter to 100% and notices that people have to occupy the same location in space in order to transmit the disease.
Comparing results of a model	The student compares the	The student expresses

run with predictions	results of their model testing with their predictions for the model's behavior at either the agent or aggregate level.	surprise that the agents in their model behave differently from their expectations, or that the aggregate-level outcomes represented by numerical readouts or graphs roughly match their mental-math predictions.
Evaluating model-run outcomes	The student evaluates the outcome of the model.	The student may announce that the outcome was boring, interesting, or surprising.
Explaining the cause of a model-run outcome	The student explains the outcome of the model at the aggregate-level by reasoning through the agent-level behavior or interactions.	For example, a student might explain that Ebola disappears very quickly in their model, leaving almost everyone healthy, because the probability of death is so high that the initially infected people die before they have a chance to infect others in the

		world.
Comparing the modeled phenomenon with other phenomena	The student predicts or makes sense of an outcome run by comparing the modeled phenomenon with another, related phenomenon.	The student compares their model of the flu to a model of Ebola.

Table 4. Refining Moves

Move	Description	Example
Noticing a problem	The student notices something problematic about the model, either in its outcome or in a piece of model code.	The student notices that an element of code, such as “move” is missing from their “go” procedure.
Modifying code to solve a problem	The student debugs the model by modifying the agent-rules or a parameter.	If the disease doesn’t spread beyond a single person, the student modifies the probability of infection of the disease, or the probability of

		recovery or death for the people in the world.
Modifying parameter values for ease of mental math	The student adjusts parameter values so that they can easily make sense of changes in value that occur in the simulation.	The student sets the number of people in the world who are initially sick and healthy so that they add up to 100, as changes in numbers of sick and healthy people over time can then be thought of as changes in percentage.
Modifying code to simplify the model	The student removes a block of code that is complicating the model.	The student has a block for "reproduce" in their "go" procedure, but then removes the block to simplify the model and understand the relationship between probability of recovery and the rate of the spread of disease.
Debugging thinking	The student debugs their thinking as a result of an	The student might initially think that Ebola is very

	unexpected model-run outcome.	deadly at the population level, however, when they run their model Ebola doesn't spread to very many people, violating their expectations. This causes them to refine their thinking about Ebola, understanding that a disease that is highly deadly for an individual is in fact less deadly for a population, because it "burns out" quickly and therefore has less of a chance to spread.

Table 5. Applying Moves

Move	Description	Example
Describing the outcome of a model run	The student describes the aggregate-level phenomenon or outcome of the model run.	The student exclaims that all of the sick people disappeared very quickly.
Interpreting numerical	The student interprets the	The student traces their finger

readouts	graph to understand the aggregate-level data.	along the curve and notes that the population of sick people is increasing and then decreasing over time.
Coordinating data from multiple readouts	The student coordinates data from different readouts in the interface.	The student looks at the readout for the number of people who died to make sense of the graph that shows the percentage of people who are sick over time.
Referencing data	The student refers to data as evidence when making a claim about the phenomenon.	The student refers to numbers on the graph to talk about how the population of sick people has increased or decreased over time.
Making sense of outcomes	The student makes sense of a model run's outcome by considering the causal variables at play or by reasoning about the causal processes encoded in the	For example, the student might determine that probability of infection is a key variable at play in their model of disease spread.

	model.	
Explaining the aggregate-level phenomenon as a result of agent interactions	The student explains the aggregate-level phenomenon as a result of agent interactions.	The student explains that the number of people who are sick increases slower at first because there are fewer people to spread the infection.
Comparing the model with the real world	The student compares the results of the model run with their understanding of the phenomenon in the real world.	The student compares their model of the flu with the Spanish flu epidemic of the early 20th century.
Comparing the modeled phenomenon with other phenomena	The student compares the phenomenon they have modeled with another, related phenomenon.	The student compares the deadliness of Ebola at the population level, to the deadliness of the flu.
Drawing conclusions about complex systems dynamics	The student uses the model to draw conclusions about complex systems dynamics, including the emergent nature of aggregate-level	The student notes that Ebola is less dangerous to a population than the flu, or that epidemics are hard to start.

	phenomena, the non-linear dependency of system outcomes on parameters, and the importance of feedback and thresholds.	
Assessing reasonableness of results	The student assesses the reasonableness of the results of their model.	The student notes that the number of people who get sick makes sense, given the probability of recovery.
Looking for key relationships	The student notes a key element of theory that they would like to identify.	The student thinks there is a ratio between recovery rate and probability of infection.

Table 6. Meta-knowledge

Move	Description	Example
Identifying limitations of the modeling environment	The student identifies limitations with the modeling environment.	The student notes that there are no blocks that would allow one to write the program so that the healthy people move around the

		world while the sick people stay at home.
Distinguishing critical from cosmetic components	The student distinguishes between components of the model that are critical, vs. those that are merely cosmetic.	The student changes the indicator of sick vs. healthy people from different colors to different shapes, but notes that this change won't have any impact on the outcome of the model run.
Noticing the approximate nature of the model	The student notes the approximate nature of the model.	The student notes that the model is different from the real world in terms of the number of people in the world.
Identifying how the approximate nature of the model may or may not impact model outcomes	The student identifies ways the approximate nature of the model may or may not impact the result of its simulation.	The student notes that a city with a million people which has an initial population of sick people of 1% has many more sick people than a world with the same percentage sick but only 200 people, and that

		this may impact whether or not the disease takes off.
Awareness of the limits of one’s own knowledge	The student notes an awareness of the limits of their own knowledge.	The student admits they don’t know what the probability of infection is in real life for both the flu and Ebola.
Reaching for credible resources	The student reaches for credible resources in making choices about initial parameter values.	The student navigates to the website of a well-known and respected organization like the World Health Organization, rather than a blog, when looking up infectiousness of Ebola vs. flu.