Journal of Science Education and Technology

The Computational Thinking for Science (CT-S) Framework: Operationalizing CT-S for K-12 Science Researchers and Educators

--Manuscript Draft--

Manuscript Number:	JOST-D-21-00314
Full Title:	The Computational Thinking for Science (CT-S) Framework: Operationalizing CT-S for K-12 Science Researchers and Educators
Article Type:	Original Research
Keywords:	computational thinking; computational thinking for science; activity theory; K12 science education
Funding Information:	division of research on learning in formal and informal settings (1838992) — Dr. Rena Dorph
Abstract:	Contemporary science is a field that is becoming increasingly computational. Today's scientists not only leverage computational tools to conduct their investigations, they often must contribute to the design of the computational tools for their specific research. From a science education perspective, for students to learn authentic science practices, students must learn to use the tools of the trade. This necessity in science education has shaped recent K-12 science standards including the Next Generation Science Standards (NGSS) which explicitly mention the use of computational tools and simulations. These standards, in particular, have gone further and mandated that computational thinking be taught and leveraged as a practice of science. While computational thinking is not a new term, its inclusion in K-12 science standards has led to confusion about what the term means in the context of science learning and to questions about how to differentiate computational thinking from other commonly taught cognitive skills in science like problem-solving, mathematical reasoning, and critical thinking. In this paper, we propose a definition of Computational Thinking for Science (CT-S) and a framework for its operationalization in K-12 science education. We situate our definition and framework in a theoretical framework from the learning sciences, Activity Theory, in order to position computational thinking as an input to and outcome of science learning that is mediated by computational tools.
Corresponding Author:	Tim Hurt University of California Berkeley Oakland, CA UNITED STATES
Order of Authors:	Timothy Hurt Eric Greenwald Sara Allan Matthew A. Cannady Ari Krakowski Lauren Brodsky Melissa A. Collins Ryan Montgomery Rena Dorph
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	University of California Berkeley
Corresponding Author's Secondary Institution:	
First Author:	Timothy Hurt

First Author Secondary Information:	
Order of Authors Secondary Information:	
Suggested Reviewers:	Richard Lamb East Carolina University His research interests focus on the identification and measurement of cognitive processes engaged while using technology in the learning of science and other STEM fields which is directly relevant to this paper.
	Irene A Lee Massachusetts Institute of Technology Lee's research that focuses on students' development of computational thinking skills has informed our development of, and thinking around, our CT-S framework.
	Marcia C. Linn University of California Berkeley Linn's background in cognition and the learning sciences and focus on science and technology matches with our use of Activity Theory to discuss technology use in science education.
	Eric Wiebe North Carolina State University We believe Dr. Wiebe will have valuable insights regarding our framework based on his work on the integration of computational thinking practices into K-16 STEM instruction.
	Keisha Varma Dr. Varma would have a useful lens with which to review our paper due to her work that focuses on learning sciences and cognition in technology-enhanced classroom settings.

The Computational Thinking for Science (CT-S) Framework: Operationalizing CT-S for K-12 Science Researchers and Educators

Timothy Hurt,* Eric Greenwald, Sara Allan, Matthew A. Cannady, Ari Krakowski, Lauren Brodsky, Melissa A. Collins, Ryan Montgomery, and Rena Dorph

Lawrence Hall of Science, University of California, Berkeley

Author Note

Timothy Hurt: https://orcid.org/0000-0003-3385-6377; correspondence to: thurt@berkeley.edu

Eric Greenwald: https://orcid.org/0000-0002-7966-6950

Sara Allan: https://orcid.org/0000-0002-8003-2269

Matthew A. Cannady: https://orcid.org/0000-0002-7256-3131

Ari Krakowski: https://orcid.org/0000-0003-0554-1669

Melissa A. Collins: https://orcid.org/0000-0003-1904-0822

Ryan Montgomery: https://orcid.org/0000-0002-6672-6432

Rena Dorph: https://orcid.org/0000-0003-4022-3789

Declarations

Funding

This work was supported by the National Science Foundation under Grant No. 1838992.

Conflicts of Interest/Competing Interests

The authors have no financial or non-financial interests to disclose that are relevant to the contents of this article.

Availability of Data and Material

Not applicable.

Code Availability

Not applicable.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1838992. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We thank the members of our Expert Panel — Matthew Berland, Debra Bernstein, Marie Bienkowski, Cynthia D'Angelo, Kemi Jona, Leilah Lyons, Tapan Parikh, Jennifer Wang, David Webb, Michelle Honda Wilkerson, and Marcelo Worsley — as well as Richard Correnti, Neal Finkelstein, Michael Horn, Amanda Peel, Christian Schunn, Carissa Romano, Vasiliki Laina, and Ying-Fang Chen for valuable discussions and feedback that contributed to the ideas presented in this material

The Computational Thinking for Science (CT-S) Framework: Operationalizing CT-S for K-12 Science Researchers and Educators

Abstract

Contemporary science is a field that is becoming increasingly computational. Today's scientists not only leverage computational tools to conduct their investigations, they often must contribute to the design of the computational tools for their specific research. From a science education perspective, for students to learn authentic science practices, students must learn to use the tools of the trade. This necessity in science education has shaped recent K-12 science standards including the Next Generation Science Standards (NGSS) which explicitly mention the use of computational tools and simulations. These standards, in particular, have gone further and mandated that *computational thinking* be taught and leveraged as a practice of science. While computational thinking is not a new term, its inclusion in K-12 science standards has led to confusion about what the term means in the context of science learning and to questions about how to differentiate computational thinking from other commonly taught cognitive skills in science like problem-solving, mathematical reasoning, and critical thinking. In this paper, we propose a definition of Computational Thinking for Science (CT-S) and a framework for its operationalization in K-12 science education. We situate our definition and framework in a theoretical framework from the learning sciences, Activity Theory, in order to position computational thinking as an input to and outcome of science learning that is mediated by computational tools.

Keywords: computational thinking, computational thinking for science, activity theory, K12 science education

Introduction

Computation has become critical to an ever-broadening list of disciplines, particularly within STEM (science, technology, engineering, and mathematics) fields (Kaczmarczyk & Dopplick, 2014). In the field of science, computational tools have long been employed to conduct research with greater precision, accuracy, and efficiency than would otherwise be possible. Computational tools have also enabled new modes of investigation, analysis, and explanation (Grover & Pea, 2013; Wing, 2010). Further, advances in computation have led to fundamental shifts in how scientific research is conducted, with computational tools expanding the epistemological problem space of scientific inquiry, enabling scientists to investigate “grand challenges” in science (Denning, 2017; Foster, 2006; Wilson, 1989). Put generally, as new technologies are developed, new applications of those technologies lead to new questions for investigation (Grover & Pea, 2013).

The importance of computational tools in science is not limited to adult practitioners — computational tools are becoming increasingly common in science classrooms. Just as they have advantages for scientists, the use of computational tools has been shown to support the learning of science content (Basu et al., 2013; Blikstein & Wilensky, 2009; Dickes & Sengupta 2013; Eisenberg, 2002; Grover, Pea, & Cooper, 2015; Malone, Schunn, & Schuchardt, 2018; National Research Council, 2011; Sengupta et al., 2013; Redish & Wilson 1993; Sherin, 2001; Wilensky & Reisman, 2006; White, 1993) and help students understand modern scientific practices (Driver, Leach, & Millar, 1996; Foster, 2006; Lehrer & Schauble, 2006; Malyn-Smith et al., 2018; National Research Council, 2007; Weintrop et al., 2016; Wiese & Linn, 2021). However, for students to be able to wield these computational tools effectively for science requires that they have some understanding of computation, as well as how computation can be leveraged to support a science goal (Grover & Pea, 2013). We define *computational thinking for science* as the cognitive processes involved in building or modifying a mental model of a computational tool’s functionality for the purpose of a given science activity.

Computational thinking for science does not just enable the use of computational tools as part of a science practice, it is also involved in the evaluation of these tools and in the design of new computational tools where current ones are insufficient or to solve new kinds of problems. It follows that computational thinking for science is both an input into, and outcome of, engagement with computational tools used for science and is therefore important for science teachers, learning designers, and assessment designers to attend to. To do this would require these stakeholders to understand what computational thinking for science is, the types of cognitive processes that give rise to it, and the types of science experiences where it is likely to occur. While there are numerous definitions, taxonomies, and frameworks around computational thinking and computational thinking in science, there does not yet exist a definition that is operationalizable in the ways described.

Our paper addresses this need by presenting the Computational Thinking for Science (CT-S) framework that is built off of the theoretical framework of Activity Theory. For researchers, the CT-S framework can be used to identify science learning experiences where computational thinking is likely to occur and to inform the development of assessments aimed at measuring computational thinking for science. For designers of science learning experiences and K-12 educators, the CT-S framework can support the development of learning experiences at the intersection of computational thinking and science.

Efforts at Defining Computational Thinking

The term *computational thinking* was introduced in 1980 by Seymour Papert in a discussion of the potential impacts of computers on the way people think and learn. He suggested that interactions with technology may actually contribute to the development of new types of mental processes (Papert, 1980). In 2006, Jeanette Wing reintroduced the term computational thinking into the national lexicon, aligning the term computational thinking with problem solving practices and methodologies frequently employed in the discipline of computer science. Wing argued that all students could benefit from learning how to think like a computer scientist. This led to a growing body of literature exploring ways to introduce computational thinking in K-12 education, especially by integrating computational thinking into core subjects (Jona et al., 2014; Lee, Martin, & Apone, 2014; Settle, et al., 2012; Yadav, Hong, Stephenson, 2016).

Despite growing awareness of its importance, computational thinking has been elusive to define and operationalize. Reflecting the concept's roots in computer science, definitions of computational thinking often reference methods and conventions commonly employed by computer scientists—for instance "solving problems like a computer scientist" or "approaching tasks like a programmer" (Barr & Stephenson, 2011; National Research Council, 2010). To operationalize these definitions, a definitive list of the computer science practices associated with computational thinking was needed. In 2010, the National Research Council convened a group to study the scope and nature of computational thinking. A resulting report identified over 20 practices that computational thinking could include (National Research Council, 2010). In the years since, several competing efforts to further synthesize the literature on computational thinking and propose a reduced list of key practices associated with computational thinking have led to diverse results—though there remains lack of consensus in the field about how to define computational thinking and how to distinguish computational thinking from other cognitive processes like logical reasoning and critical thinking (Brennan & Resnick, 2012; Grover & Pea, 2013; Rutstein, Snow, & Bienkowski, 2014). A review of the literature revealed that effectively incorporating computational thinking into K-12 science education requires a conceptualization of computational thinking that meets the following needs: 1) extends beyond computer science contexts, 2) describes the cognitive processes involved in computational thinking, and 3) includes thinking elicited by certain types of interactions with computational tools.

Computational Thinking beyond Computer Science Contexts

Although the term originated within the discipline of computer science, computational thinking is not synonymous with computer science, computer literacy, or programming (Bell, Andreae, & Lambert, 2010; Brinda, Puhlmann, & Schulte, 2009; Computing at School Working Group, 2012; Grover & Pea, 2013; Selby & Woollard, 2013). Conceptualizing computational thinking beyond computer science is essential in K-12 learning environments. There have been several notable efforts to identify the practices associated with computational thinking within K-12 core subjects. For example, Barr and Stephenson (2011) proposed a framework that illustrates how computational thinking concepts such as *abstraction* and *parallelization* could be incorporated into all K-12 subject areas. Malyn-Smith and colleagues (2018) as well as Dong and colleagues (2019) have introduced frameworks that can support educators in identifying opportunities to engage students in computational thinking within disciplinary learning. Others have emphasized specific subject areas. For instance, Weintrop and colleagues (2016) and Lee and Malyn-Smith (2020) have proposed frameworks for integrating practices related to computational thinking into K-12 STEM subjects. Such frameworks are valuable because they identify concrete activities that could promote computational thinking in K-12 classrooms. However, these frameworks do not provide corresponding definitions of computational thinking, as they focus on the skills or activities *related to* computational thinking rather than computational thinking itself.

Computational Thinking as a Cognitive Process

Computational thinking is a cognitive process—not an application of knowledge or of a technique (Grover & Pea, 2013; Li et al., 2020; Selby & Woollard, 2013). Accordingly, a viable definition of computational thinking must center on the concept of thought processes. A widely-employed definition of *computational thinking* meeting this criterion was proposed by Cuny, Snyder, and Wing (2010): “Computational thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent” (p. 1). An advantage of this definition is its focus on abstraction: in order to reformulate problems to communicate them to information-processing agents, students must first distill the problem into its core elements. In a review of the foundational literature on computational thinking, Selby and Woollard (2013) identified that abstraction is more than just a practice related to computational thinking; it is commonly described as a core component of computational thinking itself (Selby & Woollard, 2013). Indeed, Wing (2008) herself described abstraction as the “cornerstone” of computational thinking. An additional affordance of the definition proposed by Cuny, Snyder, and Wing (2010)—and variations of this definition, which center on formulating problems to yield a computational solution—is that it is easy to operationalize due to their precision (Krugel &

Hubwieser, 2018; Lowe & Brophy, 2017). Wing (2010) emphasizes that engaging in computational thinking does not inherently require programming knowledge. However, this definition does necessitate that one already has at least a basic understanding of the rules governing communication with the given information processing agent. For instance, to effectively express a problem in a form that would yield a computational solution, one would need to at least know that computers cannot comprehend instructions unless they are discrete and ordered (Bell, Rosamond, & Casey, 2012; Lealdino Filho, & Mercat, 2018). Accordingly, when defined this way, computational thinking is confined to interactions with computational tools with which one already has some prior experience or knowledge. The thinking that takes place during one's first encounters with a computational tool, for instance, may not necessarily be included within this conceptualization.

Computational Thinking from Interactions with Computational Tools

Interacting with a computational tool can engender critical thinking about the tool itself in addition to thinking about the problem or goal for which the tool is being employed (Ackermann, 1996; Jonassen, 2000; Kaptelinin & Nardi, 2006; Kuutti, 1996; Jonassen & Rohrer-Murphy, 1999). This is particularly true of certain types of interactions with computational tools—such as engaging with a computational tool for the first time, employing computational tools in new ways or for a new purpose, or envisioning new computational tool functionality. It is essential to develop students' abilities to think about the functionality and positionality of computational tools within an activity—particularly in science learning contexts (Ah-Nam & Osman, 2017; Grover & Pea, 2013; Sengupta et al., 2013; Shaffer & Clinton, 2006; Weintrop et al., 2016). However, these types of cognitive processes extend beyond that which is included within definitions of computational thinking that are centered on communicating instructions to an information processing agent. Whereas delivering instructions is a unidirectional transfer of information—from the computational thinker to the computational tool—these types of interactions with a computational tool elicit a bi-directional exchange of information between the computational thinker and the tool (Csizmadia, Standl, & Waite, 2019; Nardi, 1996; Solvie & Kloek, 2007). That is, one actually learns about the computational tool as a result of interacting with it. Therefore, a broader definition of computational thinking that includes the type of thinking that emerges from certain types of interactions with computational tools is needed.

In this paper, we operationalize this broader definition of computational thinking, focused on cognitive processes that occur during engagement with computational tools, within the Computational Thinking for Science (CT-S) Framework.

Theoretical Framework

In order to construct a definition of Computational Thinking for Science (CT-S) that attends to the mediating role of a computational tool in cognition, we draw on Activity Theory. Developed by Russian theorists Lev Vygotsky, Aleksei Leont'ev, and Alexander Luria, Activity Theory is a framework for investigating human behavior, understood as goal-directed activity in a specific socio-cultural setting (Engeström, 1987; Vygotsky, 2012). Our work builds on prior work using Activity Theory to examine technology-centered learning (e.g., Barab, Schatz & Scheckler, 2004; Blin, 2004; Brine & Franken, 2006; Issroff & Scanlon, 2002; Murphy & Rodriguez-Manzanares, 2008). Kaptelinin and Nardi (2006) argue that the use of Activity Theory directs attention away "from the computer as the focus of interest" allowing for the examination of "technology as part of the larger scope of human activities" (p. 5). Specifically, Activity Theory positions intelligence as being distributed throughout an activity system: knowledge and meaning-making emerge from a subject's interactions with tools and with others, rather than being created and held entirely by the subject (Pea, 1993). In using Activity Theory to frame our analysis, we understand computational thinking as situated in activity and distributed across the actors and tools involved in the activity (Engeström, 1987; Greeno, 1998). The intrapersonal cognitive processes we aim to describe in this analysis is but a piece of that broader goal-directed activity, but one that is critical to understand and describe (Greeno, 2015).

The basic unit of analysis in modern Activity Theory is the **activity system**. The activity system is viewed from the perspective of a **subject,** a person or group who serves as the primary actor. The efforts of the subject are motivated by and directed toward an **object**. Defined by Leont'ev (1978 as cited by Kaptelinin and Nardi 2006) as an activity's "true motive" (p. 139), the object can be thought of as an *activity's goal*. Imagine, for example, a science teacher leading a lesson about Newton's Law of Universal Gravitation. In this activity system, the science teacher is the subject. The teacher's goal is to elucidate gravitational force. In Activity Theory, goals are transformed into **outcomes**—results of the activity. Engeström (1987) clarifies that outcomes may be out of the subject's direct control and can even be unintended. In our example, an outcome may be that the teacher successfully communicated ideas about gravitational force to their students. An unintended outcome might be that the students developed misconceptions about gravity.

Drawing from prior work in child development, Vygotsky (1978) devised the concept of **mediation,** which has become a central concept in Activity Theory. He observed that when humans interact with the environment, they do so indirectly through the employment of mediating artifacts, or tools. **Tools** can be material or immaterial artifacts (e.g., a hammer or a theoretical framework). For instance, the science teacher in our example may use their understanding of the zone of proximal development to determine which ideas about gravity to communicate next to their students. The teacher might also employ tangible items, such as basketballs, to demonstrate gravity. The teacher may likewise use symbols, gestures, or diagrams to illustrate concepts visually. Regardless of materiality, all of these mediating artifacts are considered tools. Moreover, tools can, and often do, mediate an activity even if the subject is not consciously aware of their presence or intentionally employing them. In this example, the teacher's activity might be mediated by the teacher's learned instructional strategies, the teacher's own mental model of gravitational force, or the teacher's conceptions about student learning. Notably, tools do not need to be unique to the specific activity; many of the aforementioned tools could operate in several other activity systems.

The resulting outcome of the activity would be the product of this indirect interaction, or mediation, between the subject (the teacher) and the object (elucidating gravitational force) through the use of tools. Figure 1 depicts these relationships within a single-subject activity system.

In addition to the components of an activity system, Activity Theory includes assumptions about the nature of relationships among components. Notably, in activity systems, a subject does not merely wield tools in order to accomplish a goal—a one-way direction of influence. Activity systems are dialectically structured, such that components of the system are mutually dependent and influence one another. When a subject is using a tool to work towards a goal, it is not just the goal or the tool that is impacted; the subject is affected as well. The subject may learn something new about the tool or come to new understandings about the goal. This meaning-making process may then inform the subject's decisions and actions. In turn, these decisions and actions impact the tool and the goal. This dialectal exchange repeats until an outcome is produced.

Fig. 1 A Single Subject Activity System

Our analysis of computational thinking for science focuses on a subject's interaction with a computational tool toward a goal of creating a mental model of that tool's functionality with respect to its use in a science activity. For example, a learner (the subject) interacting with a simulation (the tool) to understand how to use the simulation to learn more about the real-world (the goal). As a result, we explore activity systems viewed from the perspective of a single subject as the unit of analysis. Moreover, while we share the perspective of cultural-historical activity theory (Engeström, 1999) which posits that the social, cultural, and historical context in which a subject acts will also mediate the cognition that emerges from that subject's activity in important ways, our focus in defining the cognitive processes involved in CT-S centers us on the mediated interactions among subjects, tools, and objects in science activity.

Context

This section details the crucial processes and decisions that led to the CT-S framework we present. This work is part of a larger study to understand CT-S as both an input to and outcome of science learning, and is situated within the Authors' Lab, which offers a theoretical framework regarding *science learning activation* that accounts for both the proximal (near-term) and potential distal (long-term) outcomes of science learning that happens both in- and out-of-school (Authors, 2016).

To ground our work, we sought to develop a conceptual framework that describes CT-S. Such a framework would enable us to (1) delineate subconstructs that specify cognitive processes representative of CT-S; and (2) operationalize CT-S subconstructs to develop an instrument to measure CT-S. A review of the extant work in this area revealed several frameworks that defined subconstructs related to computational thinking (Bienkowski et al., 2015; College Board, 2019; Google for Education, 2019; K12CS, 2019) as well as computational thinking in science (National Research Council, 2012; Weintrop et al., 2016). Though there was considerable overlap among the subconstructs in these frameworks, they lacked information on the cognitive demands and processes that typify CT-S, thus providing us little traction for operationalizing the identification and measurement of CT-S. That is, beyond knowing which practices are likely to engage students in CT-S, we needed a testable model for *how* those practices engage students in CT-S. To this end, we interrogated the existing subconstructs and their relationships in order to define a set that would meet the following criteria:

- Each subconstruct should:
    - Be distinct
    - Pertain specifically to science
    - Elicit computational thinking

Our team, with input and feedback from experts and advisors with relevant expertise (STEM educational research, learning design, computer science and computational thinking, and assessment design) underwent this synthesis and distillation process, to generate the CT-S framework presented here.

The resulting CT-S framework is laid out as a table to depict CT-S as the intersection of cognitive processes being used for science activities. With this structure, each subconstruct, or cell, in the CT-S framework is distinct, pertains to science, and elicits computational thinking. The cognitive processes represented by the column headers on the table are those that result from specific types of interactions with computational tools: reflective use, design, and evaluation. The categories of science activity that are listed by the row headers are those where computational tools are often leveraged in science: data collection, data processing, modeling, and problem-solving. These four categories of science activity do exist in other disciplines but, for the CT-S framework, we are only considering those activities as *science activities* when engagement occurs in service to a science goal.

Computational Thinking for Science (CT-S) Framework

Presented in Figure 2, the Computational Thinking for Science (CT-S) framework is intended to identify — and delineate — the CT-S subconstructs that can be used to inform the design of instructional sequences and assessments that promote or measure CT-S learning, respectively.

Fig. 2 The Computational Thinking for Science (CT-S) Framework

The CT-S framework is a table containing twelve cells, created by the intersection of four rows and three columns. The rows of the CT-S framework represent four categories of science activity (data collection, data processing, modeling, and problem-solving) where computational tools are likely to be leveraged[1] in K-12 science learning. The columns represent three interactions with computational tools (reflective use, design, and evaluation of computational tools) that give rise to the cognitive processes that depend upon computational thinking. Each cell within the CT-S framework, therefore, represents CT-S as the intersection of a row with a column. That is, any time an individual engages in a science learning experience or conducts a scientific investigation that can be categorized by one, or more, of the cells in the CT-S framework, they are engaging in Computational Thinking for Science (CT-S).

Defining Computational Thinking

The CT-S framework is built off of a definition for computational thinking (below) that centers on cognition that occurs during engagement with: **computational tools**. From Activity Theory, an artifact is considered a tool when the subject uses it as they work towards a goal. If the subject uses a tool in a way that leverages its computational affordances, then the tool is deemed a *computational* tool. Anything that can compute, or carry out sequences of arithmetic, or logical operations, automatically in accordance with a well-defined model (e.g., an algorithm) has computational affordances (e.g., digital and analog artifacts like calculators and slide rules respectively have computational affordances).

Important to note, an artifact with computational affordances can be a non-computational tool when someone uses it to work towards a goal without leveraging its computational affordances. For instance, using a calculator as a paperweight is still using the calculator as a tool but not as a *computational* tool. Whether or not a tool is a computational tool in a given use depends on its

[1]While each of these activities occur in domains other than science, our definition draws on prior work articulating the science discipline-specific instantiations of each activity.

functionality in that use — what it does to help the subject work towards the goal in a given activity system.

With this understanding of a computational tool in mind, we give the following definition:

Computational thinking is the cognitive processes involved in building or modifying a mental model of a computational tool's functionality.

Defining Computational Thinking for Science (CT-S)

Computational Thinking for Science (CT-S) occurs when an individual engages in computational thinking for their science activity. In the subsections that follow, we present three hypothetical cases of CT-S to illustrate how a student can engage in each of the three cognitive processes (reflective use, design, and evaluation of a computational tool) during a science activity. We define the three cognitive processes as follows:

- *Reflective Use of a Computational Tool:* building or modifying a mental model of that computational tool's functionality through interaction[2] with that tool.
- *Design of a Computational Tool*: building or modifying a mental model of an imagined[3] computational tool's functionality.
- *Evaluation of a Computational Tool:* building or modifying a mental model of the affordances and limitations of that computational tool's functionality.

These definitions are grounded in Activity Theory which stipulates that cognition occurs through the use of tools towards a goal. In other words, each of the above definitions assumes that the cognitive processes are happening within a goal-directed activity. In the case of CT-S, the goal-directed activity is necessarily a science activity.

Hypothetical Cases of CT-S Activity

Context for the Following Hypothetical Cases

In order to illustrate how science activities and cognitive processes intersect for our definition of CT-S, we explore three hypothetical cases of CT-S: *reflective use* of a computational tool for *data processing*, *design* of a computational tool for *data collection*, and *evaluation* of a

[2]Note that the interaction does not need to be direct. That is, an individual could instruct their friend to interact with the tool and so long as the friend communicates their actions and the computational tool's behavior, the individual could still be engaged in Reflective Use.

[3]Imagined, in this instance, refers to the fact that the subject is generating a new-to-them mental model of a computational tool's functionality regardless of whether that computational tool's functionality currently exists in the world.

computational tool for *modeling*. In each of the three hypothetical cases, a student is engaged in a scientific investigation. The science goal of that student's investigation is to better understand phenomena related to bacterial growth. While there are many different ways a student could conduct an investigation with this same goal, each example will illustrate one way the student could engage in CT-S as they work towards their science goal. These examples will narrowly focus on *how* the student engages in CT-S during their science activity. However, it is crucial to understand that their engagement in CT-S is only a step towards achieving their goal. To achieve their goal would likely involve other tools and other forms of cognition than those discussed in the cases.

Reflective Use of a Computational Tool for Data Processing

Imagine that a student is studying bacterial growth where their science goal is to learn how the bacterial population changes over time. They have data on the bacterial population size at different times that has been loaded into a graphing calculator. This student knows that graphing the data may help them to identify a relationship between bacterial population size and time; however, this student has never previously used a graphing calculator. Before this student could use the graphing calculator, they would need to figure out how to operate it and what computations it can do that may help them toward their science goal. They can do this by engaging in Reflective Use of the graphing calculator as they interact with it. The student might begin by manipulating the calculator, selectively pressing certain buttons and observing the results of those actions. They can then reflect on their manipulations and begin to form a mental model of the graphing calculator's functionality. As the student continues to interact with this computational tool, their discoveries may reinforce, revise, or supplement their developing mental model. In this way, Reflective Use is bi-directional in terms of information transfer: the student takes actions, the student then reflects on what the computational tool does as a result of those actions, and the student then takes new actions based on the result of that reflection. Through this continued engagement in Reflective Use, the student would have built a mental model of the graphing calculator's functionality and how it can help them towards their science goal. In building this mental model, the student engaged in CT-S (Figure 3). At this point, the student can use their mental model alongside the graphing calculator to process the bacterial growth data so that it is in a form that they can use to learn how the bacterial population changes over time. For example, the student may create a scatterplot using the graphing calculator and identify that the bacteria population grows at an increasing rate over time.

Fig. 3 Reflective Use of a Computational Tool for Data Processing Activity System

Reflective Use can also occur if the student already has an incomplete or inaccurate mental model of a computational tool's functionality. For instance, when the student faces an unexpected output or error, they may engage in Reflective Use of the computational tool to reinvestigate and modify their mental model of its functionality.

Reflective Use stands in contrast to *rote use* of a computational tool—wherein the student employs the graphing calculator alongside a mental model of its functionality, or by following an external script, for their data processing. For instance, imagine that the student already knew how to create graphs using the graphing calculator. If the student proceeded to use this computational tool to plot the data, this would be considered rote use, not Reflective Use (Figure 4). Reflective Use only occurs when the student interacts with the tool in a way that leads them to build or modify their mental model of its functionality. When the student engages in rote use, their mental model of the computational tool's functionality is not modified so the student does not engage in computational thinking.

Fig. 4 Rote Use of a Computational Tool for Data Processing Activity System

Design of a Computational Tool for Data Collection

Imagine that a student is studying bacterial growth where their science goal is to learn how the bacteria population grows over time. They do not have any data that they can analyze to meet their science goal, so they decide to collect the data that they will need. They have a new population of bacteria on a petri dish and they determine that they will need to collect data on the size of the bacteria population at different times. They think that the population might grow quickly — go from being invisible to filling the petri dish over the course of one day. They decide that they will use a camera that is connected to a computer to take measurements throughout the day. To plan out their data collection, the student engages in Design where they imagine a computational tool's functionality that would enable them to take a picture of the petri dish every 10 minutes for 24 hours. The student thinks about how the camera and the computer could be set up to take pictures of the petri dish at a 10-minute time interval, and the student determines that as long as the computer saves all of the images, and all of the corresponding timestamps, it will be possible to analyze the images to determine how large a given bacteria population is at each timestep. As a result of this line of thinking, the student would have built a mental model of this imagined computational tool's functionality and how it could be leveraged in their science activity; the student engaged in CT-S (Figure 5). If the student were to go on to run this experiment, the student would be able to determine if there are any relationships between the bacterial population size and the amount of time it has had to grow.

Fig. 5 Design of a Computational Tool for Data Collection Activity System

Importantly, a different student in the same science activity could engage in Design in a notably different way. For instance, this different student may know that computer camera software can analyze images and determine the relative sizes of objects in an image. This different student might Design a computational tool for data collection where the camera remains on, like a video camera, but only saves images, and timestamps, when one of the objects in the video stream has changed in size by a specified amount.

It is important to note that, because computational thinking is a form of cognition, an individual can engage in Design *without* physically or digitally constructing their *imagined* computational tool's functionality. For example, the outcome of Design in the previous paragraph did not include the actual programming of the data collection device. Another important aspect of Design is that it need not only precede the creation of a computational tool's functionality nor occur only once in a creation process. That is, Design can occur throughout an iterative creation process where the developer has to repeatedly update and modify their mental model of the computational tool's functionality relative to its intended use to support their goal.

Evaluation of a Computational Tool for Modeling

Imagine that a student is studying bacterial growth where their science goal is to learn how the bacteria population grows over time in different conditions. They want to know if a simulation they found online is an accurate model of the real world that they can use for their investigation. The student decides to engage in Evaluation of the simulation to determine its affordances and limitations with respect to their science goal. To engage in Evaluation, the student needs to know what the simulation should do in different configurations to determine if it is indeed an accurate model. To do this, the student may conduct some research to determine what they should be comparing the simulation's results to. For instance, they may find that bacterial growth curves tend to exhibit distinct phases depending on certain factors, like time elapsed, nutrient concentration, and species. Based on this research, the student determines that the simulation does a good job at modeling growth if a population has unlimited resources, but since the model doesn't include the ability to control resources, it is unable to model a growth curve when nutrients are limited. As a result, the student would have built a mental model of the affordances and limitations of the computational tool's functionality and how it could be leveraged in their science activity, the student engaged in CT-S (Figure 6). Having completed this Evaluation, the student could then determine whether to use the simulation as they work toward their science goal. For instance, they may use the tool simply to investigate growth with unlimited resources and what it looks like in a bacterial population or they may select a different computational tool based on the result of their Evaluation of that tool relative to their science goal.

Fig. 6 Evaluation of a Computational Tool for Modeling Activity System

In this example, the student completed their Evaluation based on research and a mental model of the simulation that they built through Reflective Use. It is important to note that Evaluation could also work if a student used a mental model of a computational tool's functionality that they had designed. Similarly, the student could use their existing knowledge of the underlying phenomenon in their Evaluation instead of conducting research.

Discussion

The definition of CT-S that we propose was borne out of a need to operationalize the construct so that it could be accurately and reliably measured. In addition to further testing of its use for that purpose, we see a need for further research and theoretical work that can apply this definition to ground the design of learning experiences (e.g., how can tasks be designed to provide students practice with CT-S in ways likely to advance learning), program evaluation (e.g., to examine how well activities are aligned with goals, and goals with observable outcomes), as well as for policy initiatives and funding decisions (e.g., predicting what set of initiatives are most likely to lead to desired outcomes).

In calling for this additional research, we also recognize that we have thus far examined computational thinking for science without explicit attention to the cultural-historical mediators of activity characteristic of modern Activity Theory (Engeström, 1999). With measurement development as the primary motivator of our work, we made this choice in order to focus attention on the individual contributions of the learner within the activity: what are the cognitive resources brought to bear in tool-mediated, goal-oriented activity and how might those resources, those mental models, become visible and get revised through activity. A complete understanding of CT-S within an activity system, however, must attend to the complex situativity of learning represented in the "bottom row" of the modern Activity Theory triangle: the rules, community, and divisions of labor within the activity system (Engeström, 1999). Our hope is that the present focus on understanding the subject—tool—object activity system does not obscure or contradict our broader commitment to understanding learning, and CT-S, as situated activities. In this commitment, we share Sengupta and colleagues' call that "computing and computational thinking should be viewed as discursive, perspectival, material and embodied experiences, among others. These experiences include, but are not subsumed by, the use and production of computational abstractions." (Sengupta, Dickes, & Farris, 2018, p.49). Our argument is that it is around the use and production of computational abstractions where CT-S is most visible and offers the most tractable location to ground the design of learning experiences.

At the same time that we see CT-S as situated in activity, we see the activity of CT-S as situated within the "mangle of practice" (Pickering, 2010) that characterizes knowledge construction in science. As summarized by Sengupta and colleagues, "scientists struggle continuously in order to get theories and instruments on one hand and the natural world on the other to perform in the ways that their investigations require." (2018, p52) Appreciating this requires appreciating the complex ways CT-S is engaged by the learner and their peers, as well as by the professional scientist and their colleagues. For example, the hypothetical cases of Reflective Use, Design, and Evaluation presented above were provided to help illustrate CT-S concretely and simply. Because of this, the examples provided did not illustrate any potential distal outcomes of engaging in CT-S. It is important that such outcomes be considered, even if they are not a requirement of CT-S, as they are often cited as a reason to promote CT-S within science and science education. For example, take a student who is engaging in the Design of a simulation of a real-world system. As they consider the parameters to include in their simulation, they may realize that they do not actually know how to model one of the relationships within their simulation such that it would accurately reflect the real-world system. This would likely lead them to research the real-world relationship until they are satisfied that they could model it correctly in their simulation. In this example, while it was not initially a goal of their activity, in order to continue working on their Design, they determined that they needed to increase their knowledge about a specific real-world phenomenon. This example illustrates how CT-S can motivate science learning beyond engagement with the computational tool. As a second example, we will illustrate how CT-S can motivate science learning beyond the initial science goal while still focusing on the engagement with the computational tool. Imagine a student is using a simulation to study predator-prey relationships. As they are engaged in rote use, they notice a menu option that allows them to modify the relative speeds of the predators and prey. As they enter into Reflective Use they start asking new questions that go beyond their original science goal. After modifying their mental model of the simulation's functionality, they engage in a use of the simulation that helps them learn science beyond their original science goal. This example illustrates how CT-S Reflective Use can provide opportunities for students to ask and investigate new science questions. It also reveals one way in which computational tools developed through and for scientific research have enabled new scientific discoveries that were otherwise non-investigable: in much of modern science, scale, complexity, and observability limitations are mediated by computational tools that calculate, model and simulate natural phenomena in novel and transformative ways, enabling old problems to be solved and new questions to be asked. CT-S is inextricably wrapped up in the practice of modern science, and its isolation for the purposes of measurement or instructional design should not imply its severability from other scientific practices *in vivo*.

One test of this CT-S framework will be its potential usefulness in examining how the mangle of practice within science intersects with that of computer science, a field where computational abstractions are the principle outcomes of activity as well as necessary mediating artifacts, and where programming knowledge is the coin of the realm. While we reject an interpretation of CT-S that requires programming knowledge, we anticipate variation in how one engages in CT-S according to one's programming knowledge[4]. Concretely, one who has a certain level of programming knowledge could engage in CT-S differently from one who does not, yet both

[4] All of the examples of CT-S given in this paper have excluded the act of computer programming or coding in order to illustrate that a student does not *have* to code or even know how to code in order to engage in CT-S.

could still engage in CT-S. For example, in each of the modalities of CT-S, the student who is engaged in CT-S must have some knowledge of, or have made certain types of assumptions about, the computational tool with which they are engaged. As we posit that CT-S is a form of cognition that arises through engagement with computational tools in science-motivated activity, it is important that we consider programming knowledge, for instance, as a separate artifact that could mediate activity for a subject. Our treatment of science activities as integral to the CT-S framework (Figure 2) is our attempt to operationalize this complexity of practice. However, further development and scrutiny of measures and of the designs of learning experiences grounded within this framework are necessary to examine how useful this attempt at operationalizing CT-S will be. We posit that an operationalizable CT-S framework will advance research and practice in science learning and propel efforts to position the experiences of individual computational thinkers within their situational learning contexts.

Ethical Statement

Not Applicable

Consent Statement

Not Applicable

References

Ackermann, E. (1996). *Constructionism in practice: Designing, thinking, and learning in a digital world.* Routledge.

Ah-Nam, L., & Osman, K. (2017). Developing 21st century skills through a constructivist-constructionist learning environment. *K-12 STEM Education, 3*(2), 205-216. The Institute for the Promotion of Teaching Science and Technology (IPST). https://www.learntechlib.org/p/209542/

Authors. (2016).

Barab, S., Schatz, S., & Scheckler, R. (2004). Using activity theory to conceptualize online community and using online community to conceptualize activity theory. *Mind, Culture, and Activity*, *11*(1), 25-47. https://doi.org/10.1207/s15327884mca1101_3

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?. *ACM Inroads, 2*(1), 48-54. https://doi.org/10.1145/1929887.1929905

Basu, S., Dickes, A., Kinnebrew, J., Sengupta, P., & Biswas, G. (2013). CTSiM: A computational thinking environment for learning science through simulation and Modeling. *CSEDU*. https://doi.org/10.5220/0004390103690378

Bell, T., Andreae, P., & Lambert, L. (2010, January). Computer science in New Zealand high schools. *Proceedings of the Twelfth Australasian Conference on Computing Education-Volume 103* (pp. 15-22). https://doi.org/10.1145/2157136.2157240

Bell, T., Rosamond, F., & Casey, N. (2012). Computer science unplugged and related projects in math and computer science popularization. *The multivariate algorithmic revolution and beyond* (pp. 398-456). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-30891-8_18

Bienkowski, M., Snow, E., Rutstein, D., & Grover, S. (2015). Assessment design patterns for computational thinking practices in secondary computer science: A first look. *SRI International*. https://pact.sri.com/resources.html

Blikstein, P., & Wilensky, U. (2009). An atom is known by the company it keeps: A constructionist learning environment for materials science using agent-based modeling. *International Journal of Computers for Mathematical Learning*, *14*(2), 81-119. https://doi.org/10.1007/s10758-009-9148-8

Blin, F. (2004). CALL and the development of learner autonomy: Towards an activity-theoretical perspective. *ReCALL: the Journal of EUROCALL*, *16*(2), 377. https://doi.org/10.1017/S0958344004000928

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 Annual Meeting of the American Educational Research Association (AERA)*, Vancouver, Canada (Vol. 1, p. 25).

Brine, J., & Franken, M. (2006). Students' perceptions of a selected aspect of a computer mediated academic writing program: An activity theory analysis. *Australasian Journal of Educational Technology*, *22*(1). https://doi.org/10.14742/ajet.1305

Brinda, T., Puhlmann, H., & Schulte, C. (2009). Bridging ICT and CS: educational standards for computer science in lower secondary education. *ACM Sigcse Bulletin, 41*(3), 288-292. https://doi.org/10.1145/1595496.1562965

College Board. (2019). *AP Computer Science Principles.* AP Course Overview. https://apcentral.collegeboard.org/pdf/ap-computer-science-principles-course-overview.pdf?course=ap-computer-science-principles

Computing at School Working Group. (2012). Computer Science: A curriculum for schools. *Computing at School Working Group*. https://www.computingatschool.org.uk/data/uploads/ComputingCurric.pdf

Csizmadia, A., Standl, B., & Waite, J. (2019). Integrating the Constructionist Learning Theory with Computational Thinking Classroom Activities. *Informatics in Education, 18*(1), 41-67. https://doi.org/10.15388/infedu.2019.03

Cuny, J., Snyder, L., & Wing, J. M. (2010). Demystifying computational thinking for non-computer scientists. [Unpublished manuscript]. https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Denning, P. J. (2017). Computational thinking in science. *American Scientist, 105*(1), 13-17. https://www.doi.org/10.1511/2017.124.13

Dong, Y., Catete, V., Jocius, R., Lytle, N., Barnes, T., Albert, J., Joshi, D., Robertson, R., & Andrews, A. (2019, February). PRADA: A practical model for integrating computational thinking in K-12 education. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 906-912). https://doi.org/10.1145/3287324.3287431

Driver, R., Leach, J., & Millar, R. (1996). *Young people's images of science.* McGraw-Hill Education (UK).

Eisenberg, M. (2002). Output devices, computation, and the future of mathematical crafts. *International Journal of Computers for Mathematical Learning*, *7*(1), 1-44. https://www.doi.org/10.1023/A:1016095229377

Engeström, Y. (1987). *Learning by expanding: An activity-theoretical approach to developmental research* [Doctoral dissertation].

Engeström, Y. (1999). Activity theory and individual and social transformation. *Perspectives on activity theory*, *19*(38), 19-30. https://doi.org/10.1017/CBO9780511812774.003

Foster, I. (2006). A two-way street to science's future. *Nature, 440*(7083), 419-419. https://doi.org/10.1038/440419a

Google for Education. (2019). *CT Overview*. Exploring Computational Thinking. https://edu.google.com/resources/programs/exploring-computational-thinking/#!ct-overview

Greeno, J. G. (1998). The situativity of knowing, learning, and research. *American psychologist*, *53*(1), 5. https://doi.org/10.1037/0003-066X.53.1.5

Greeno, J. G. (2015). Commentary: Some prospects for connecting concepts and methods of individual cognition and of situativity. *Educational psychologist*, *50*(3), 248-251. https://doi.org/10.1080/00461520.2015.1077708

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38-43. https://doi.org/10.3102/0013189X12463051

Grover, S., Pea, R., & Cooper, S. (2015, April). Systems of assessments for deeper learning of computational thinking in K-12. *Proceedings of the 2015 Annual Meeting of the American Educational Research Association (AERA),* Chicago, IL (pp.15-20).

Issroff, K., & Scanlon, E. (2002). Educational technology: The influence of theory. *Journal of Interactive Media in Education*, *6*. http://doi.org/10.5334/2002-6

Jona, K., Wilensky, U., Trouille, L., Horn, M. S., Orton, K., Weintrop, D., & Beheshti, E. (2014, January). Embedding computational thinking in science, technology, engineering, and math (CT-STEM). *Future Directions in Computer Science Education Summit Meeting*.

Jonassen, D. H. (2000). Revisiting activity theory as a framework for designing student-centered learning environments. *Theoretical foundations of learning environments*, 89-121.

Jonassen, D. H., & Rohrer-Murphy, L. (1999). Activity theory as a framework for designing constructivist learning environments. *Educational Technology Research and Development, 47*(1), 61-79. https://doi.org/10.1007/BF02299477

K12CS. (2019). *K–12 Computer Science Framework.* https://k12cs.org/wp-content/uploads/2016/09/K%E2%80%9312-Computer-Science-Framework.pdf

Kaczmarczyk, D., & Dopplick, R. (2014). *Rebooting the pathway to success: Preparing students for computing workforce needs in the United States.* Education Policy Committee. Association Computing Machinery. https://pathways.acm.org/ACM_pathways_report.pdf

Kaptelinin, V., & Nardi, B. A. (2006). Acting with technology: Activity theory and interaction design. MIT press.

Krugel, J., & Hubwieser, P. (2018). Strictly objects first: A multipurpose course on computational thinking. *Computational Thinking in the STEM Disciplines* (pp. 73-98). https://doi.org/10.1007/978-3-319-93566-9_5

Kuutti, K. (1996). Activity theory as a potential framework for human-computer interaction research. *Context and Consciousness: Activity Theory and Human-computer Interaction*, 1744.

Lealdino Filho, P., & Mercat, C. (2018). Teaching computational thinking in classroom environments: A case for unplugged scenario. *Proceedings of the Resources 2018 International Conference* (pp. 296-299).

Lee, I., & Malyn-Smith, J. (2020). Computational thinking integration patterns along the framework defining computational thinking from a disciplinary perspective. *Journal of Science Education and Technology, 29*(1), 9-18. https://doi.org/10.1007/s10956-019-09802-x

Lee, I., Martin, F., & Apone, K. (2014). Integrating computational thinking across the K—8 curriculum. *ACM Inroads, 5*(4), 64-71. https://doi.org/10.1145/2684721.2684736

Lehrer, R., & Schauble, L. (2006). *Cultivating model-based reasoning in science education*. Cambridge University Press.

Leont’ev, A. N. (1978). *Activity, consciousness, and personality*.

Li, Y., Schoenfeld, A. H., diSessa, A. A., Graesser, A. C., Benson, L. C., English, L. D., & Duschl, R. A. (2020). Computational thinking is more about thinking than computing. *Journal for STEM Education Research*, 1–18. Advance online publication. https://doi.org/10.1007/s41979-020-00030-2

Lowe, T., & Brophy, S. (2017, October). An operationalized model for defining computational thinking. *2017 IEEE Frontiers in Education Conference (FIE)* (pp. 1-8). https://doi.org/10.1109/FIE.2017.8190682

Malone, K. L., Schunn, C. D., & Schuchardt, A. M. (2018). Improving conceptual understanding and representation skills through Excel-based modeling. *Journal of Science Education and Technology*, 27(1), 30-44.

Malyn-Smith, J., Lee, I. A., Martin, F., Grover, S., Evans, M. A., & Pillai, S. (2018). Developing a framework for computational thinking from a disciplinary perspective. *Proceedings of the International Conference on Computational Thinking Education* (p. 5).

Murphy, E., & Rodriguez-Manzanares, M. A. (2008). Using activity theory and its principle of contradictions to guide research in educational technology. *Australasian Journal of Educational Technology*, *24*(4). https://doi.org/10.14742/ajet.1203

Nardi, B. A. (1996). *Context and consciousness: Activity theory and human-computer interaction.* MIT Press.

National Research Council. (2007). *Taking science to school: Learning and teaching science in grades K-8*. National Academies Press.

National Research Council. (2010). *Report of a workshop on the scope and nature of computational thinking.* National Academies Press.

National Research Council. (2011). *Learning science through computer games and simulations*. National Academies Press.

National Research Council. (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*, 65-66. National Academies Press. https://www.nap.edu/catalog/13165/a-framework-for-k-12-science-education-practices-crosscutting-concepts

Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas.* Basic Books, Inc.

Pickering, A. (2010). *The mangle of practice: Time, agency, and science*. University of Chicago Press.

Redish, E. F., & Wilson, J. M. (1993). Student programming in the introductory physics course: MUPPET. *American Journal of Physics*, *61*(3), 222-232. https://doi.org/10.1119/1.17295

Rutstein, D. W., Snow, E., & Bienkowski, M. (2014, April). Computational thinking practices: Analyzing and modeling a critical domain in computer science education. *Proceedings of the 2014 Annual Meeting of the American Educational Research Association (AERA)*, Philadelphia, PA.

Selby, C., & Woollard, J. (2013). Computational thinking: the developing definition.

Sengupta, P., Dickes, A., & Farris, A. (2018). Toward a phenomenology of computational thinking in STEM education. *Computational thinking in the STEM disciplines*, 49-72. https://doi.org/10.1007/978-3-319-93566-9_4

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies, 18*(2), 351-380. https://doi.org/10.1007/s10639-012-9240-x

Settle, A., Franke, B., Hansen, R., Spaltro, F., Jurisson, C., Rennert-May, C., & Wildeman, B. (2012, July). Infusing computational thinking into the middle-and high-school curriculum. *Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education* (pp. 22-27).

Shaffer, D. W., & Clinton, K. A. (2006). Toolforthoughts: Reexamining thinking in the digital age. *Mind, Culture, and Activity, 13*(4), 283-300. https://doi.org/10.1207/s15327884mca1304_2

Sherin, B. L. (2001). How students understand physics equations. *Cognition and instruction*, *19*(4), 479-541. https://doi.org/10.1207/S1532690XCI1904_3

Solvie, P., & Kloek, M. (2007). Using technology tools to engage students with multiple learning styles in a constructivist learning environment. *Contemporary issues in technology and teacher education, 7*(2), 7-27. https://www.learntechlib.org/primary/p/22811/.

Vygotsky, L. S. (1978) *Mind in Society: The Development of Higher Psychological Processes*, ed.M. Cole, V. John-Steiner, S. Scribner and E. Souberman. Cambridge, MA: Harvard University Press.

Vygotsky, L. S. (2012). *Thought and language.* Cambridge, MA. MIT Press.

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. https://doi.org/10.1007/s10956-015-9581-5

White, B. Y. (1993). ThinkerTools: Causal models, conceptual change, and science education. *Cognition and Instruction*, *10*(1), 1-100. https://doi.org/10.1207/s1532690xci1001_1

Wiese, E. S., & Linn, M. C. (2021). “It Must Include Rules” Middle School Students’ Computational Thinking with Computer Models in Science. *ACM Transactions on Computer-Human Interaction (TOCHI)*, *28*(2), 1-41. https://doi.org/10.1145/3415582

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. https://doi.org/10.1207/s1532690xci2402_1

Wilson, K. G. (1989). Grand challenges to computational science. *Future Generation Computer Systems, 5*(2-3), 171-189. https://doi.org/10.1016/0167-739X(89)90038-1

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35. https://doi.org/10.1145/1118178.1118215

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366*(1881), 3717-3725. https://doi.org/10.1098/rsta.2008.0118

Wing, J. (2010). Research notebook: Computational thinking—What and why. *The Link Magazine, 6*. https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding 21st century problem solving in K-12 classrooms. *TechTrends, 60*(6), 565-568. https://doi.org/10.1007/s11528-016-0087-7

Figure 1

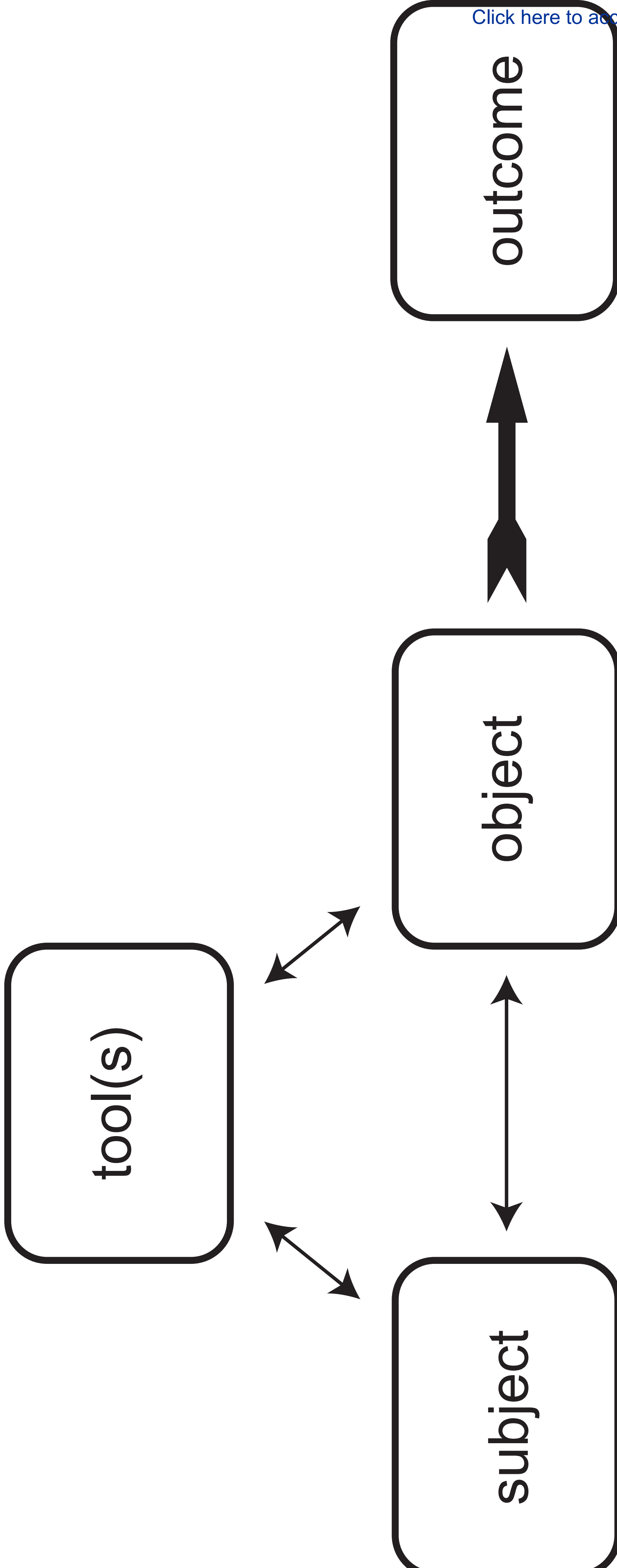


Figure 2

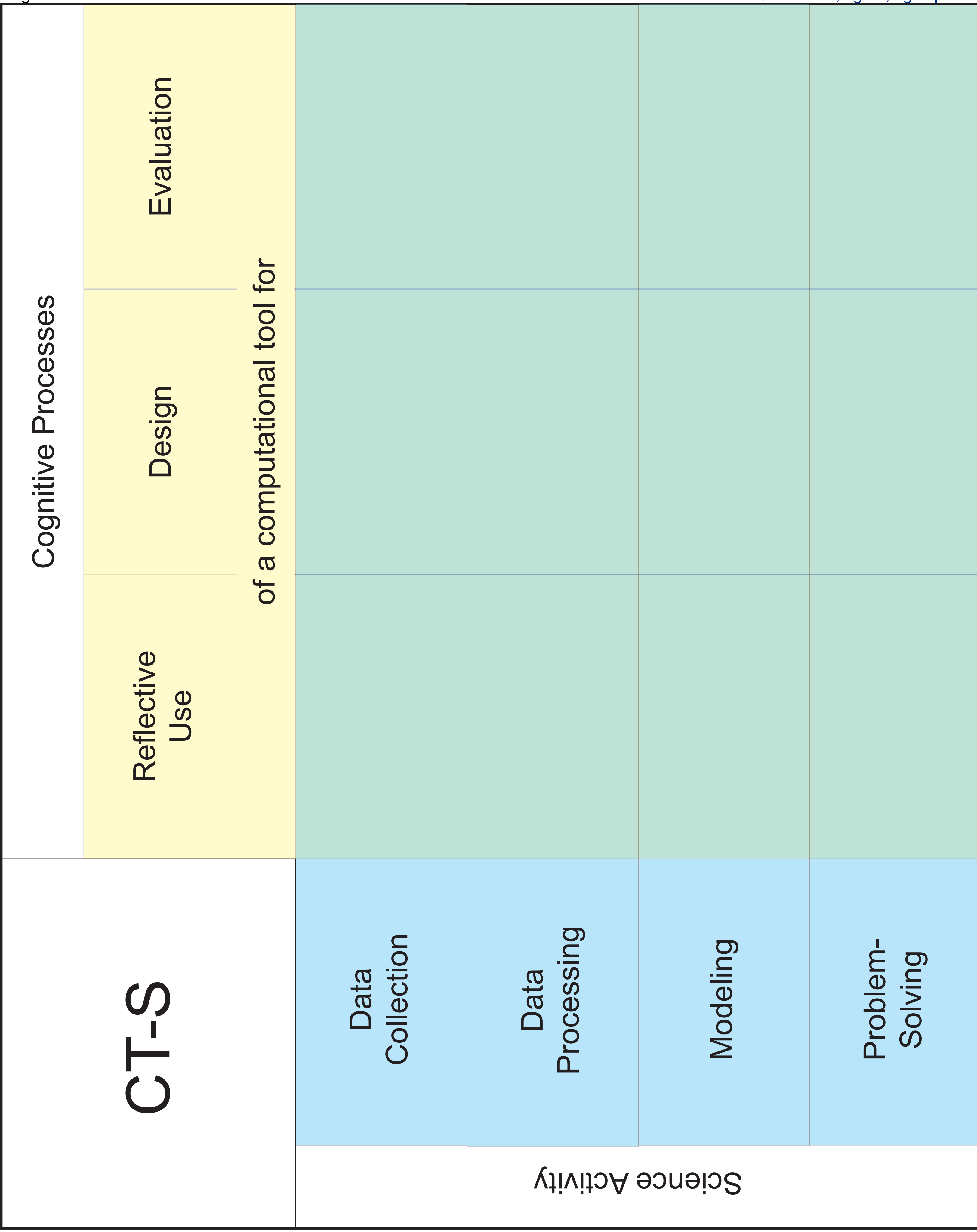

CT-S
Cognitive Processes
Reflective Use
Design
Evaluation
of a computational tool for
Science Activity
Data Collection
Data Processing
Modeling
Problem-Solving

Figure 3

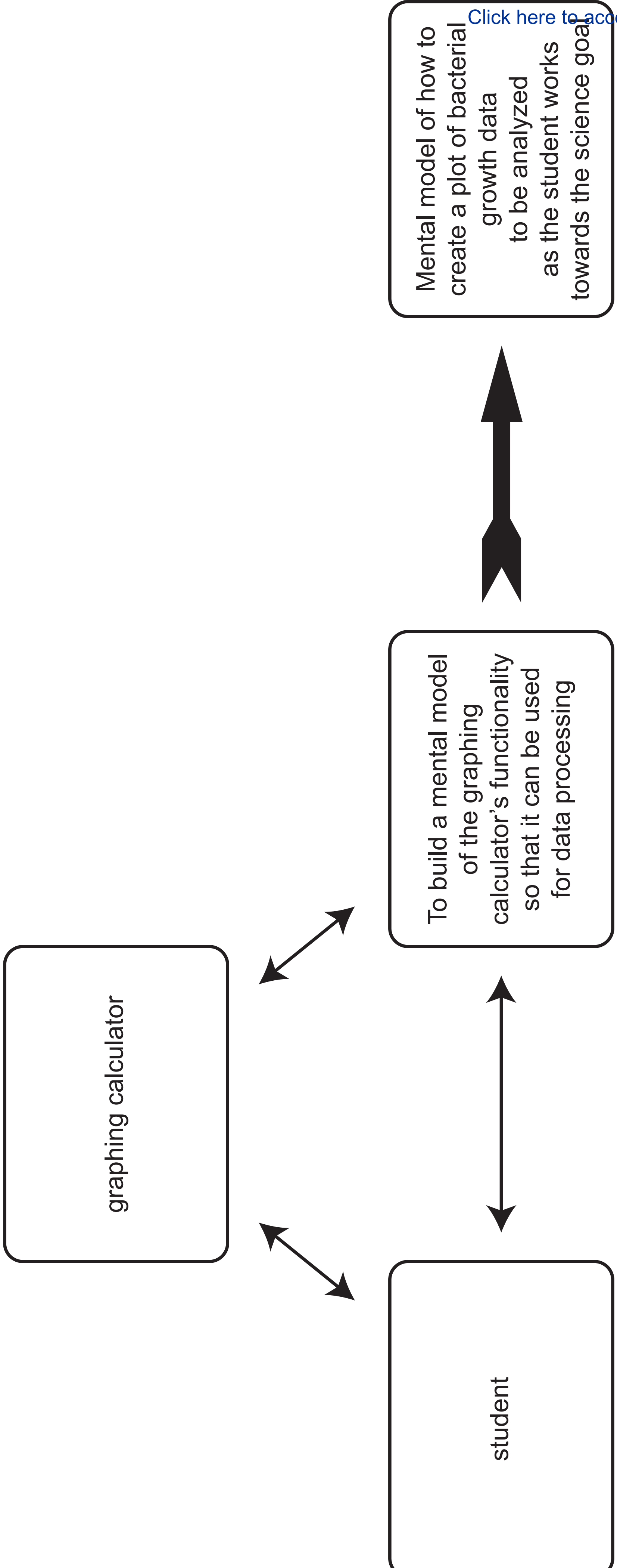

Mental model of how to create a plot of bacterial growth data to be analyzed as the student works towards the science goal
To build a mental model of the graphing calculator's functionality so that it can be used for data processing
graphing calculator
student

Figure 4

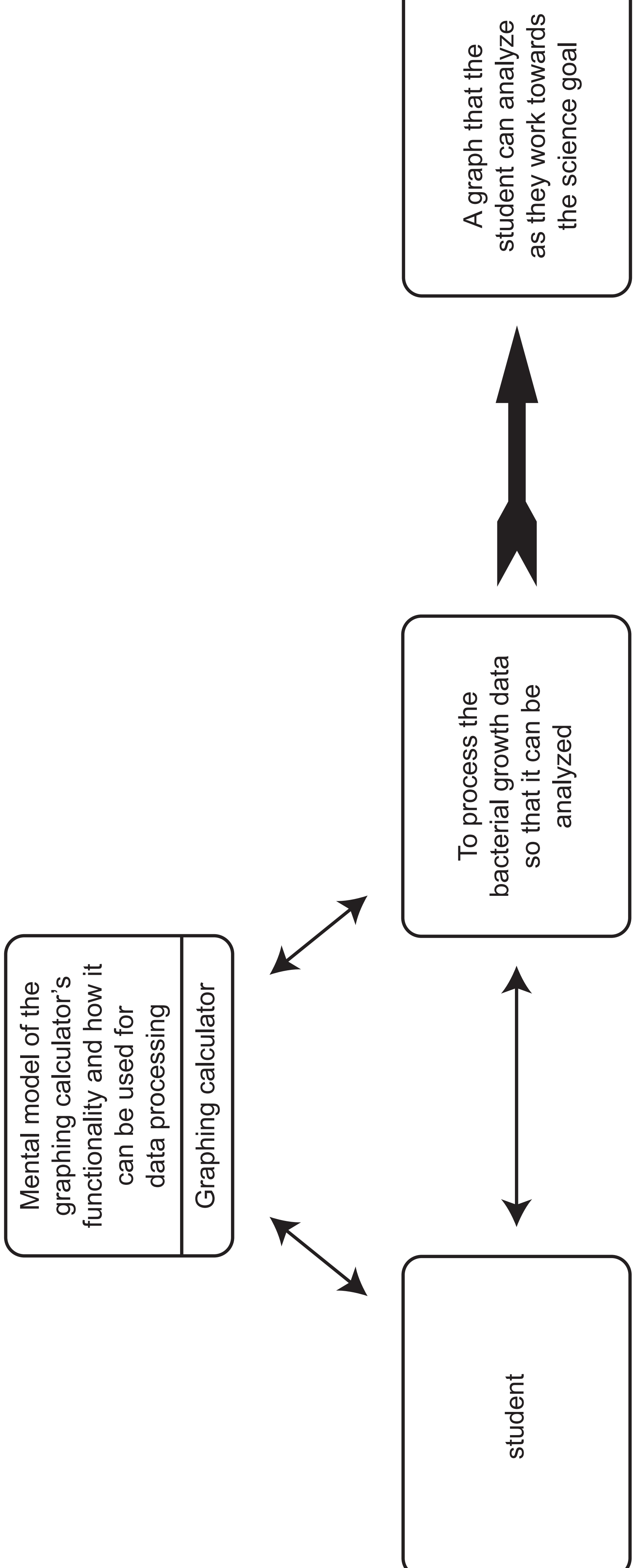

A graph that the student can analyze as they work towards the science goal
To process the bacterial growth data so that it can be analyzed
Mental model of the graphing calculator's functionality and how it can be used for data processing
Graphing calculator
student

Figure 5

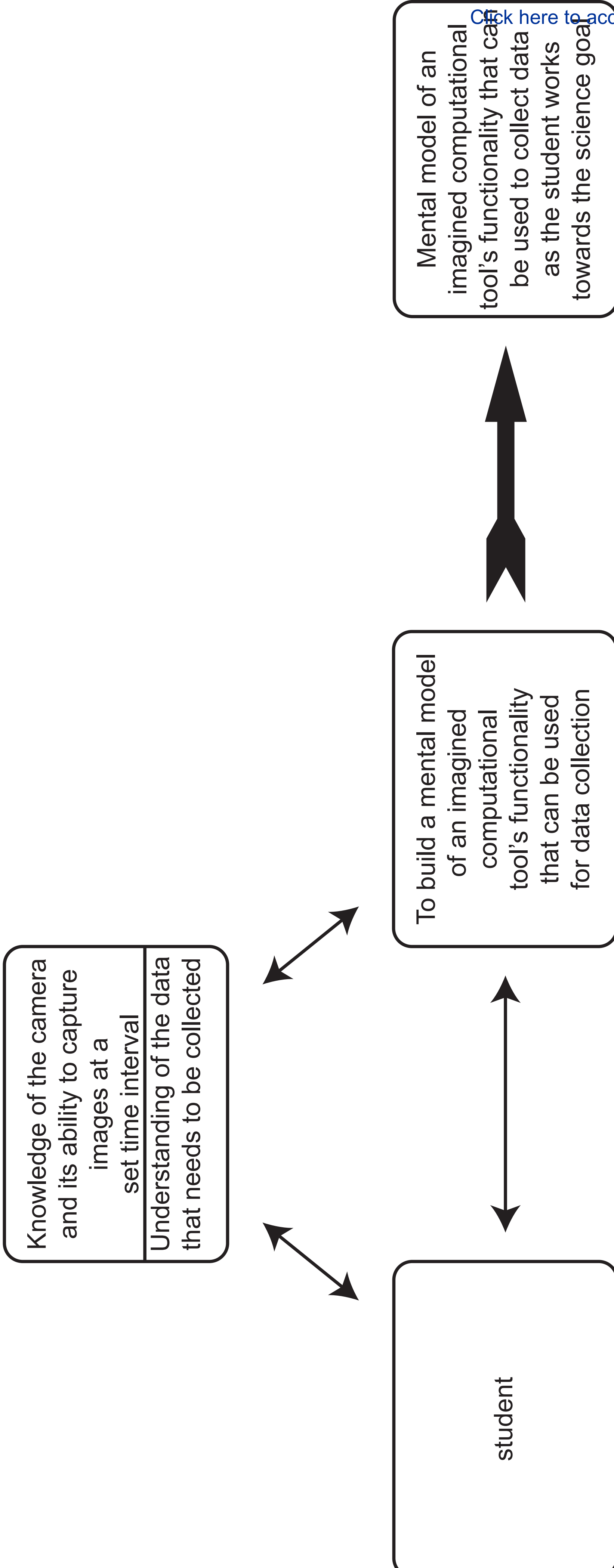
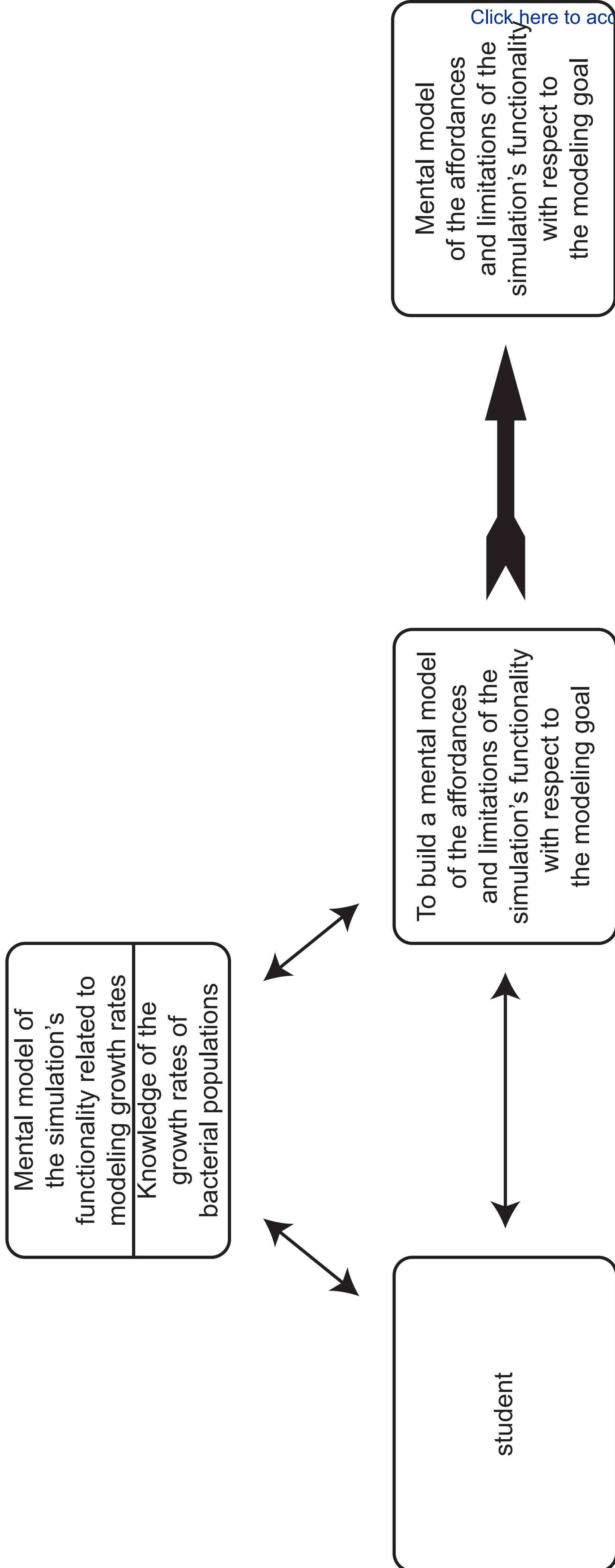

student
Knowledge of the camera and its ability to capture images at a set time interval
Understanding of the data that needs to be collected
To build a mental model of an imagined computational tool's functionality that can be used for data collection
Mental model of an imagined computational tool's functionality that can be used to collect data as the student works towards the science goal

Figure 6


student
Mental model of the simulation's functionality related to modeling growth rates
Knowledge of the growth rates of bacterial populations
To build a mental model of the affordances and limitations of the simulation's functionality with respect to the modeling goal
Mental model of the affordances and limitations of the simulation's functionality with respect to the modeling goal