Reflections on Significant Developments in Designing SimCalc Software

James Burke, Stephen Hegedus, and Ryan Robidoux

From the first SimCalc project (see timeline in introduction), researchers and software developers made decisions shaping the design of the software based on contributions from students, teachers, and an evolving understanding of what this technology made possible in the classroom. We developed the SimCalc learning environments using the principles of dynamic interactive technologies suggested by Kaput (1994).

Our development of this software over time allowed us to consider and refine its design according to observations of its use in classrooms, advances in the available technology, and evolving theories of how students can learn important mathematics. New development brought new technological affordances. Important decisions can be seen in the mathematical representations, in the configurability of the software, and in the communication infrastructure that we developed to take advantage of networked activities. Importantly, as designers and developers, we have seen technological affordances as a way to introduce new mathematical affordances into the classroom.

We focus on two main themes throughout this chapter that describe major shifts in the evolution of the SimCalc learning environment: (1) The rationale for how changes in the software were made and the decisions that drove such changes, and (2) How our thinking about the software changed in terms of new affordances for learning. We will focus on three main areas of research and how each has guided the design, development and implementation of the software: (1) Representational

J. Burke (✉) · S. Hegedus · R. Robidoux
Kaput Center for Research and Innovation in STEM Education, University of Massachusetts Dartmouth, 285 Old Westport Road, North Dartmouth, MA 02747, USA
e-mail: jburke@umassd.edu

S. Hegedus
e-mail: shegedus@umassd.edu

R. Robidoux
e-mail: ryan.robidoux@umassd.edu

S.J. Hegedus, J. Roschelle (eds.), *The SimCalc Vision and Contributions*, Advances in Mathematics Education, DOI 10.1007/978-94-007-5696-0_5,
© Springer Science+Business Media Dordrecht 2013

Infrastructures and how design is guided by mathematically meaningful representations, (2) Activity Structures, and (3) Classroom Connectivity and its impact on learning and participation. As researchers and software developers that make up the core design team, we prefer not to describe a litany of how the software changed over time but rather major changes to exemplify how our research was tightly integrated into our development by focusing on improving mathematical learning for many different kinds of learners. While the process of development took place over more than 15 years of continually-funded projects, an especially dramatic shift took place in the later years. SimCalc MathWorlds® (hereon referred to simply as SimCalc) had been a student-centered, 1:1 software environment that mediated meaningful student exploration and expression of mathematical ideas. It has become a mathematics-learning environment that connects the entire classroom, allowing the interrelationships among student contributions to be the basis of classroom-level discussion and thought.

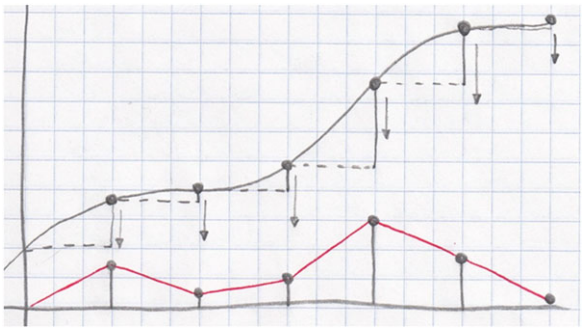
As the development cycle neared its close, the notion of the importance of getting the software into the hands of users to inform design decisions was bookended by design changes that would, in turn, put the software into the hands of more students, particularly underrepresented groups. Dissemination became a priority in order to allow wider access to important mathematical affordances for educational and research purposes. Design decisions play a role in helping the software reach a larger community.

1 Innovation Research Software Development

The SimCalc design team (including researchers, software developers, and teachers) sought not only to enhance existing classroom curriculum, but also to transform it with activities that were not possible without the use of technology. We intended for students and teachers to engage in thinking about mathematics in powerful ways. Introducing SimCalc into the classroom frequently was, for us, a means to see what might be possible with mathematics learning and instruction in ways that eventually led us to support the emergence of new forms of participation.

Hegedus and Moreno-Armella (2010) has described the co-evolution of technology and user action that draws research and development together this way as a relationship between instrumentation (i.e., a shaping of a participant's actions through co-actions with a tool) and instrumentalization (i.e., the shaping of the tool itself by a user's knowledge and by the environment). This iterative design cycle led to highly configurable software that could support a wide variety of pedagogical and curricular intentions to produce mathematically meaningful lessons. And when we sought to introduce the affordances of networked devices, the communication infrastructure introduced a new connection to mathematical structure (e.g., a family of functions). We found that SimCalc lessons could structure the physical setup of the classroom (e.g., how students were placed in groups and how the work of each group was structured) in ways related to the mathematics. Students were afforded different ways to interact because the classroom itself had become mathematically structured.

Fig. 1 Sketch of "lollipops"



2 Mathematically-Meaningful Design

2.1 Representational Infrastructure

The representations that are central to SimCalc have always been tied to thinking about how they served mathematical meaning. Even in early discussions, Kaput described his visions for representations that would connect mathematical concepts through the actions of users and through some dynamic presentation of the software. He outlined an animated method for connecting the graph of some varying quantity to a graph that approximated the change in that quantity over time. He suggested that the changes over unit time could be drawn and then animated by dropping them to the axis, where they would provide a very rough approximation of a graph of change. This idea for connecting (for example) position and velocity graphs was affectionately referred to as "lollipops" (see Fig. 1).

It was never implemented because it merely existed to serve as the beginning of a conversation not just about what we could possibly do with the available technology, but that we should always think about mathematical structure and connections as part of how we envisioned the development and use of the software as a learning environment.

2.2 Dynamically-Linked Representations

An important representational infrastructure of SimCalc is its dynamically-linked representations. This allows different "views" of the functions through the user interface that are linked so that any changes in a function are reflected in all representations (see Fig. 2). For example, the editing of a position graph is immediately and continuously reflected in a velocity graph.

The details of these representations are important to preserving them as mathematically meaningful. For example, the segments of piecewise functions in SimCalc software are considered to be open on the leftmost end and closed on the rightmost end.

In the creation of activities, many of these functions are built with segments that start and end on integer values of x. The implications of how segment boundaries are

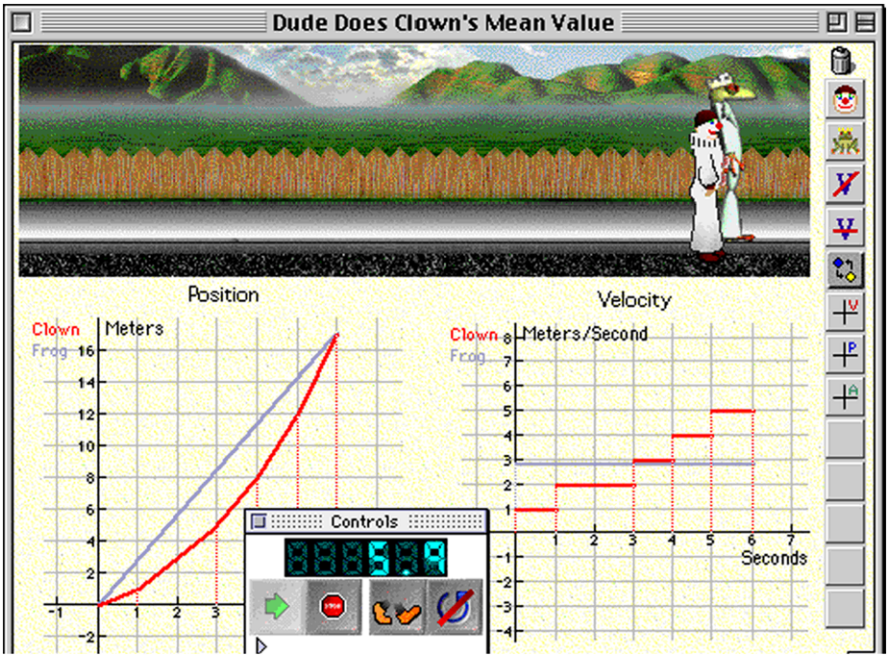


Fig. 2 Screenshot of the multiple, linked representations in Classic MathWorlds (name used prior to SimCalc MathWorlds®)

defined become apparent when a teacher makes a table to show the velocities at each integer value of x. Developing the software to display the velocity at moments when the velocity is changing instantaneously presents a problem. We resolved this problem by determining into which interval that x value falls, but the result may not be what a teacher expected. The details of the representation can be mathematically justified, but this is just one example of how ambiguity produces the need to make a decision in some seemingly minor detail of how the software works in legitimate ways.

These details are necessary, intentional, and part of the dynamic experience students can have by acting on the software. As Hegedus and Moreno-Armella (2010) note, the editing of a linear function is accomplished using very distinct types of (user) actions (and user interface elements). "Hotspots" that appear on the graph allow a student to explicitly edit the domain or range of a function through separate actions. With the editing of each variable separated, a student can see the mathematical consequences of changing one or the other (see Fig. 3).

2.3 "Ghosting"

While the idea of separate hot-spots for editing the graph representation was a core *mathematical* idea in the earliest prototypes of SimCalc software, some aspects of

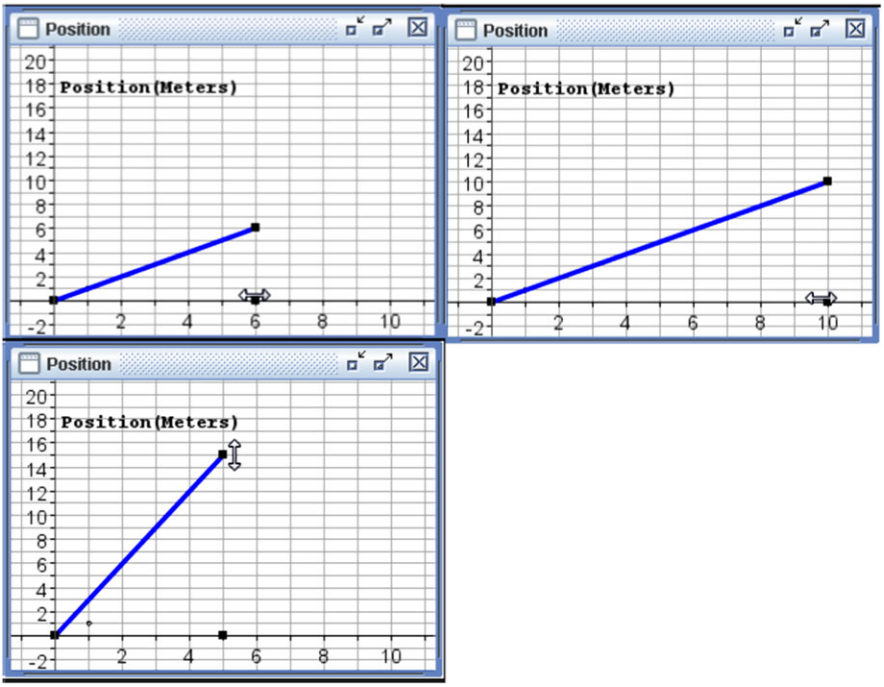


Fig. 3 This is a constant velocity position graph showing oversized hotspots used for editing the domain and range of this segment of a function. The point at (5, 0) in the *first image* can be dragged horizontally to edit domain. The point at (5, 6) in the *first image* is constrained to vertical dragging, allowing a change in the function's value at time $t = 5$, but also the slope of the function. The *other images* show the results of those two separate, possible drag actions. Note also the *cursor*, which has changed to indicate the possible drag direction

a representation developed over many iterations and was revisited in numerous discussions over extended periods of time as the software saw increasing classroom use and feedback flowed back to the project with the aim of improving student learning. One such representational detail is the "ghosting" effect of characters in the motion representation (the "world" as it is called). The idea of ghosting was created to solve the problem of using functions that were only defined over an interval or a restricted domain in an environment that animates the characters using that data. This was an important design principle that was mathematical in its intent and purpose.

In early versions of the software, it was unclear what to do with the animated characters (often referred to as "actors") when the end of a function's domain was passed during animation. Arguments can be made for a number of ways to handle this situation. A function is undefined outside of its domain, so you might represent that by having the actor appear only when the animation is within the defined time domain. The consequence of this is that actors will be appearing and disappearing while the animation clock is running. A trial of this approach proved to be confusing for students, despite its mathematically justified interpretation of "undefined."

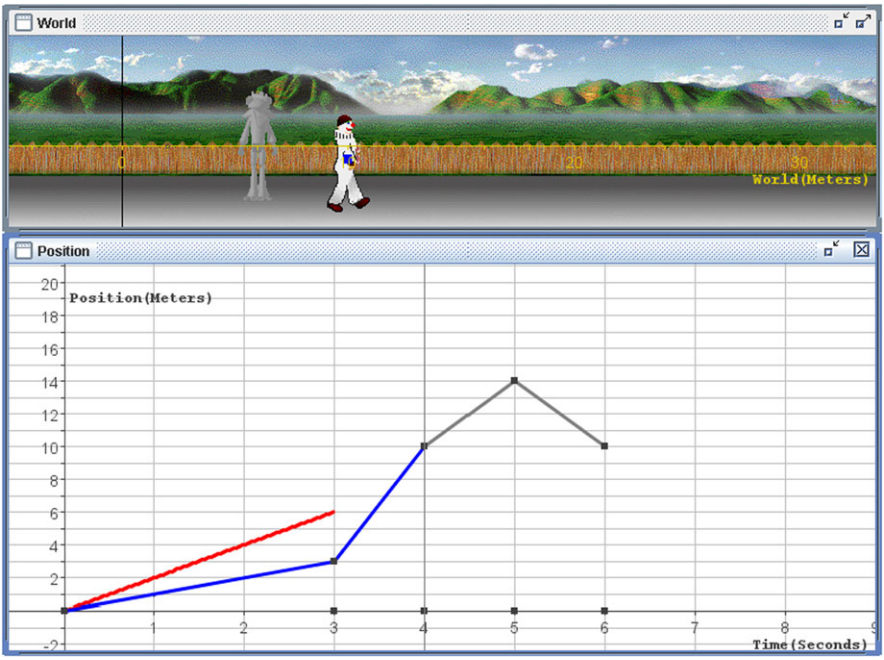


Fig. 4 An example of an actor that is "ghosted" once its domain ends at 3.0 seconds

Another approach is to have the character persist somehow. For example, the actor can simply stop at the position where its data "runs out." This simple solution is problematic because showing an actor at some position during a time when his position is undefined in the graph is inconsistent. The two representations contradict each other mathematically, and maintaining the position is arbitrary. Consider a sonar tracking system in a submarine. If you are following some sort of target and your contact with the target is interrupted, you have no current position data. It might be reasonable to assume that your target continued to move with its last known velocity instead of coincidentally stopping at its last known location. Making assumptions about missing data can depend on the context. The choice of representation should not be arbitrary.

Since many SimCalc activities involve actors getting to a specific position, students need to clearly see the end of a character's motion. This is accomplished if the actor remains at that position (rather than continue moving, as in the submarine example above). This justifies using "last known position" but does not solve the representational conflict with the graph. Our solution was to ghost the character out. Ghosting is accomplished by graying out the character's image to indicate that the data has run out, or the function of the character's motion is undefined (see Fig. 4). The additional indication allows the mathematical meaning of the end of domain to be represented even while a remnant of the actor remains as a marker; the state of being undefined is simultaneously represented while the last known position is

preserved. Among the different choices of how these dynamic representations can behave, the decisions are made to be mathematically meaningful and provide the maximum visual feedback to students in ways that provide support to the goals of classes of activities.

To resolve what was both a usability problem and a problem in mathematical integrity, we looked to the structure of activities and to enriching the representation for a solution that solved both problems.

3 Activity Structures

3.1 Activities as an Organizing Principle

In the late 1990s, when the SimCalc development team was considering how to take advantage of new software environments that could cross platforms and possibly allow new ways to envision software as being composed of reconfigurable components, the notion of what a SimCalc activity was became a focal point. The "Classic MathWorlds" application (SimCalc's Mac-only product) had existed for a few short years and had the dynamic representations for which SimCalc was already known. It also supported scripting through AppleScript (Roschelle et al., 1996), allowing other researchers to alter and augment its functionality in meaningful ways, like changing the method of student input from dragging a hotspot to one that used prompts and text input boxes so the students could enter a numerical value (Olive and Lobato, 2008).

The SimCalc team began using Sun's Java environment to create applets that would be able to deliver the core SimCalc curriculum on any web browser. An applet, so-named to imply something smaller than a full application, is a program designed to run in a web browser, usually with a limited user interface specific to a particular task. A feature of applets is their somewhat isolated nature; they could be run within a browser, but the browser environment limited the applet's access to information from the user's computer. These security-driven restrictions were limiting to developers looking to connect applets. A number of applets were created based on the combinations of necessary representations (position, velocity and acceleration graphs and different animation worlds and meters). These applets were created by, literally, disassembling the curriculum units and re-grouping the activity based on what representations (Cartesian graphs, animation world, a single number-line-like meter) were needed to complete the activity. This allowed researchers to expand the activities to multiple platforms.

There were challenging limitations in this proliferation of applets, and eventually the applet approach was abandoned. However, it had forced the design team to consider activities as an important way to think about the structure of the software, and the structure of the use of the software. Central to the activity was what

held it together as mathematically meaningful. For the applets, the use of specific representations was an observable surface attribute, but also reflected the mathematical structure of an activity: an activity that needed both an animation world and a velocity graph relied on the mathematical link between those representations. The intentions of a well-defined activity structure are reflected in how the software is configured for any given activity.

3.2 The Problem and Opportunity of Configurability

After SimCalc became a single application, configurability was revisited. There was also a sense that the increasing set of features of SimCalc allowed possibilities beyond one vision for implementation of a curriculum. Designers and developers of the software considered the possibility that it could become a generative environment for activity development and implementation.

The main concerns over configurability were considered in two categories of issues: the first having to do with opportunistic discovery of ideas that could be explored in the classroom and the second having to do with activity building. When working with students, Kaput would often remark that he had come across some significant, interesting, and surprising situation that resulted from students engaging in a discussion of some mathematical insight. It was natural that, as a researcher, he would want the ability to make a small change in the activity, based on his pedagogical expertise. He did this in order to provide the students opportunities to explore further, to extend the reach of a successful activity based on what the students had brought to the situation. This allowed Kaput to make observations about what was possible in the undergraduate classroom; it provided specific moments where the understanding of what was possible could be seen to co-evolve with the activity itself. This led to user-configurability becoming a central function of the SimCalc software so that researchers and teachers in the future could engage in similar actions as Kaput, both in the classroom and in preparation for a class.

An example of an early configurability feature issue under discussion was “inner windows.” The applet version of SimCalc and the first desktop version presented the activity with immutable representations. For example: one position graph and an animation world. Not only would a user be unable to add a velocity graph, but the relative sizes and position of these representations were fixed. When the possibility of draggable inner windows was offered, we implemented them. This introduced configurability options with significant implications for the way an activity could unfold. With draggability came (1) re-arranging representations, (2) hiding representations, and (3) opening a new representation. Re-arranging representation allowed a teacher to do such things as place a vertical animation world next to a position graph so that students could see that a height along the y-axis corresponds to a position in the animation world. Hiding a representation allowed the teacher to create a situation in which students see one representation and discuss another one

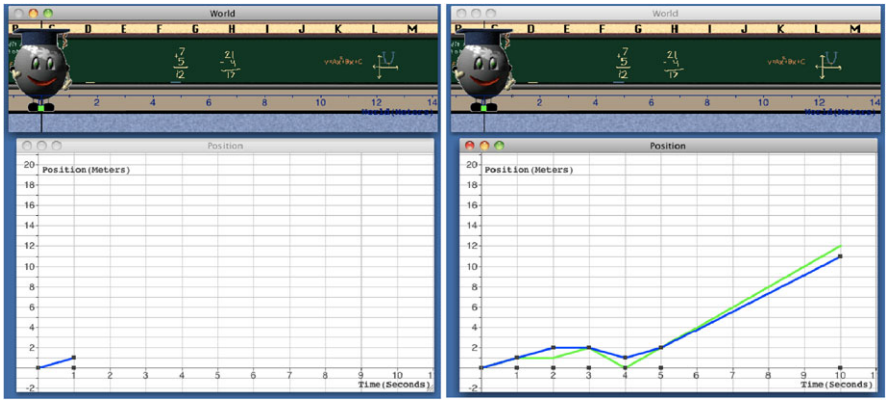


Fig. 5 On the *left* is the position graph with the (*light*) green actor's graph hidden, and, on the *right*, is the position graph where the (*light*) green actor's graph is shown along with the student-created graph, which is in (*dark*) blue. (Color figure online)

without seeing it. For instance, students watching an animation could try to describe and even argue about what the position graphs would look like. The position graph could then be revealed to spark further discussion about the predictions and encourage meaning making across multiple representations—a critical mathematical skill. Opening a new representation could allow a teacher to make a point that crosses representations to follow a student's assertion, for example. These three activity structures were enabled by a change in the control the teacher had in the configurability of the software. The second structure—hiding representations—proved to be extremely powerful. An example of this structure is illustrated in the screenshots shown in Fig. 5. On the left is the representational view at the beginning of an activity where students are asked to create a linear-piecewise position graph, shown in blue, which represents the motion of the green actor, whose graph is hidden. Once the students' graph is complete, the teacher shows the green actor's graph, as shown on the right side of Fig. 5, to begin a discussion comparing the two graphs.

Configurability brought challenges as well. The more flexibility that was built into the software, the greater the opportunity existed to mutate the activity. And while mutations can be beneficial, they can also be fatal to the intentions of the activity (Brown and Campione, 1996). Fidelity of implementation became more of an issue, and a possible argument against configurability. Configurability must recognize constraints that define the boundaries of activity within which explorations remain close to the intentions of the activity designer.

Where that boundary line is drawn was a source of recurring debate. A solution that emerged was to entrust teachers with wide latitude as authors of activities, and interpreters of an activity's intentions. Teachers were trusted to understand the intention of an activity in the belief that they are primarily focused on implementing a curriculum activity, and only secondarily interested in developing or modifying the way an activity is presented in the software.

4 Classroom Connectivity

Classroom connectivity was a major expansion of the SimCalc learning environment, connecting students and teachers through a network of computer-based devices running the software. With the advent of network connectivity, the communication infrastructure of the classroom was radically changed, as student work could now be aggregated and publicly displayed on the teacher's computer. This would impact the range of mathematical discourse across groups and individual students in the classroom. Mathematical discourse became an even more central focus for understanding how students could learn in such an environment as well as be motivated to learn more in the future as discussed further in other chapters of this book (for example, Dalton and Hegedus; and Brady, White, Davis and Hegedus, this volume) Furthermore, with these new forms of participation and communication in the classroom, new activity structures within a SimCalc classroom emerged. Not only were new activity structures designed for the affordances of connectivity, but activity structures were developed and evolved from those affordances.

4.1 Evolution of Network Connectivity Within the SimCalc Environment

The communicational infrastructure of SimCalc began as a network of graphing calculators running a pared-down version of the SimCalc computer software and evolved into a cross-platform network of personal computers running the complete, representationally-rich SimCalc environment.

The communications infrastructure emerged from efforts to use the social network within the classroom and incorporate that into the structure of an activity. Kaput had often discussed his desire to see activities that involved numerous actors in the animation world, each with a function that formed part of a family of functions that varied systematically, like a strange army whose marching revealed some mathematical variation instead of regimented uniformity. There were technical limitations that held us back from smoothly animating such an "army" of actors in the world, but those limitations lifted with time. And as Kaput's *army of animated actors* became a technical reality, the SimCalc team began to consider how students might be the ones to give that army its marching orders.

4.2 Graphing Calculators and Our Partnership with Texas Instruments

From 2000, Jim Kaput and Stephen Hegedus began to simultaneously design two versions of SimCalc software—one for the popular TI-83+ graphing calculators and

a separate cross-platform computer application that extended the work discussed earlier in this chapter.

The primary concern at the time was developing something that was affordable and utilizable on a platform that was used at scale. Texas Instruments (TI) had obtained large market penetration in the sale of graphing calculators. A portable document format (APPVAR) allowed developers to create small calculator activities that could be distributed with paper activities in TI's on-line store.

This was a challenging time for the SimCalc development team, as they had to deal with multiple programming issues particularly in deploying a fairly complex representational infrastructure in a small amount of memory. Secondly, the creation of activities was done in-house due to many complexities and the lack of a simple authoring environment making scale problematic. Nevertheless, several curriculum packages were released through the TI webstore that focused on linear functions, slope as rate and averaging problems using rate graphs. The main demand was creating a compact animation that was smooth and linked to other representations including graphs and algebraic expressions. Many compromises were made due to the screen resolution and the lack of color. The development team focused on using just two actors due to such limitations that in turn constrained the kinds of activities that could be designed. But this was a time to contrast the constraints and affordances of such devices with respect to their equivalent but more expensive computer counterparts.

At the same time there was a new dawn in connectivity. People were talking more about the potential of social networks and how such potential could be introduced into classrooms. Connecting graphing calculators through a hardware and software environment was being actively discussed at TI in consultation with the SimCalc design team and other partners.

At SimCalc headquarters in Massachusetts, development in this direction was being undertaken on the desktop application. Development was extremely new and highly prototypical, with design cycles iterating every day, as new ideas for activity structures were discussed and the software development team worked on creating a communication infrastructure using simple network protocols. The SimCalc team found a local school to partner with who had a lab of networked e-Macs. The driving force behind this groundbreaking development was the search for deploying new activity structures that modified the way mathematics was thought about in classrooms and where each student could contribute something mathematically meaningful. At the time, it was unclear how much impact such insights would have on modifying the very nature of participation in the classrooms, and how this might impact learning and motivation. Aided by funding from the National Science Foundation (NSF) in 2000[1] and 2004[2], the SimCalc design team produced a network applica-

[1]PI: Kaput, J. (2000–2003). *Understanding classroom interactions among diverse, connected classroom technologies*. REC-0087771.

[2]PI: Kaput, J. & Hegedus, S. (2004–2009). *Representation, participation and teaching in connected classrooms*. REC-0337710.

tion that worked in concert with the SimCalc application on several computers and discovered a lot about what types of activities could be done under such conditions.

5 MathWorlds Server: A Glimpse into New Forms of Student Participation

The new, connected classroom prototype was designed on a push-pull model, whereby students pushed their work (i.e., constructed functions) from their SimCalc environment to a server, and the teacher would then aggregate all student contributions within her version of SimCalc. The network topology involved teachers and students working on desktop computers running SimCalc, with an additional computer dedicated to running the SimCalc MathWorlds Server application. Once a student completed a task, she would log into the server and submit her function across the network. When teachers aggregated all student work from the server, the students' representations (e.g., motion, graphs) were projected on the publically viewed display of the teacher's computer. Prior to this prototypical version of SimCalc, activities involved operating on 2–3 functions. However, with the advent of student contributions, teachers would be operating on functions to an order of magnitude based on the number of students in their class. The curriculum and software developers realized that a new level of configurability had to be built into the SimCalc environment to handle this scale up of available functions. Thus, the View Matrix was created to allow teachers to hide and show the representations contributed by students (e.g., graphs, actor motions) in their publically displayed SimCalc environment.

In 2002, the connectivity-enhanced SimCalc environment, which combined the data collection afforded by SimCalc Server and highly interactive group-based classroom activities, was piloted in a 5-week, after-school program with 7th, 8th and 9th grade students. For the first time in the development of this communicational infrastructure, students would participate in a networked classroom over an extended period of time. During this pilot intervention, the entire SimCalc design team—researchers, software developers, teachers—descended upon the after-school program, and was witness to new forms of student participation that emerged within the connected classroom (Hegedus and Penuel, 2008).

5.1 Connected MathWorlds on Multiple Platforms

In 2003, it was clear that the development trajectories of SimCalc on the TI-83Plus and on the computer were proving to be problematic. With a new burst of funding from the NSF in 2004, Kaput and Hegedus set about re-conceptualizing both software platforms within one environment, based upon TI's newly created Navigator

product. Over the next few years, Navigator became a successful commercial product, and the SimCalc team worked closely with TI to develop a calculator application that could receive and send packets to the SimCalc application on the computer through semi-wireless networks, where the calculators would be connected to wireless hubs communicating via a proprietary Access Point connected to the teacher computer.

Still, major issues resulted from the use of graphing calculators in the SimCalc network—in terms of teacher usability and network stability. Teachers, on the whole, struggled with the setup of the classroom network, which involved configuring TI-Navigator hubs that were not controlled via the SimCalc software. Even when the Navigator network was properly setup, the communication between the teacher computer and the student calculators was prone to miscommunication. To increase reliability of the network, protocols were added to the software to handle various network states that arose due to this instability. For example, protocols were implemented to allow for students that had been dropped from the network to reconnect and receive all messages that were sent across the network during their absence. These protocols marked the first substantial effort to move the SimCalc environment towards a commercial-level of stability, and allowed the SimCalc team to research *Connected* SimCalc at scale during a longitudinal efficacy study in Massachusetts' high schools as reported in this book and elsewhere (see Dalton and Hegedus, this volume; Dalton et al., 2011).

By 2007, the SimCalc application for the TI-83/84Plus had been completely rewritten from scratch with a brand new interface, and communication as its central core. Simultaneously, the computer version of SimCalc was re-written into a Java Application with an embedded protocol utilizing TI network infrastructure. Finally, both versions were working together and co-evolving in their development cycles. The computer version became a more parent application, which could configure and send activities to the calculator, as well as collect student work. After an aesthetic overhaul, which set the computer version of SimCalc on a potential commercialization trajectory, design challenges now shifted from specific activity functions to broader functionality, such as activity configuration across both platforms, e.g., representational control (see Sect. 5.1.1 on classroom management) and activity design that could aggregate the work of up to 32 students each working on a calculator.

Funding from the US Department of Education's Institute of Education Sciences[3] allowed the SimCalc team to conduct a large efficacy study in Massachusetts which solidified this work, allowing them to shift their development to curriculum at scale and train teachers from a wide variety of backgrounds to implement it. The SimCalc team was now in a position to implement the integrated software and curriculum project into several school systems.

Such efficacy and scale-up work was not limited to calculators though, and one great result of such work is that the co-development of a product on two platforms can improve both platforms in profoundly important ways, which would not have

[3]PI: Hegedus, S. (2007–2012). *Democratizing access to core mathematics across grades 9–12*. #R305B070430.

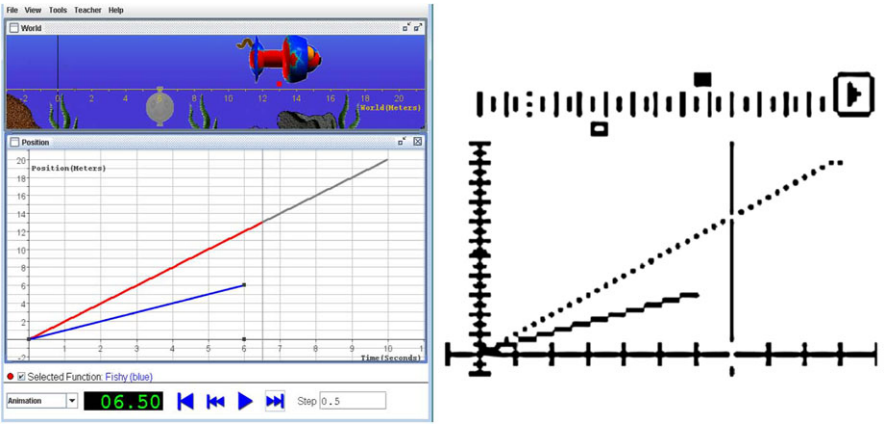


Fig. 6 Students' SimCalc environments on computers, *left*, and graphing calculators, *right*

been possible otherwise. The Java application of SimCalc was still fully functional as a stand-alone desktop application, working on both PC and Macs. This was used in a large randomized control trial in Texas (see chapters by Shecthman, Haertel, Roschelle, Knudsen and Singleton; Roschelle and Shecthman; and Vahey, Roy and Fueyo, this volume). Similarly, SimCalc on TI-83/84 calculators could work in an off-line mode if a network was not found and activities could be exported by the parent computer application for such stand-alone use.

In 2006, during a bleak period after losing Kaput, Hegedus decided to move development into a commercialization trajectory to help fuel the new Kaput Center which the University of Massachusetts had approved, and over the next 2 years both products went into a thorough testing and quality assurance phase. Many followers became part of the team to test and also to think about producing the software in multiple languages. In 2008, the first commercial version of the computer software was released with a new name that incorporated a federal trademark, SimCalc MathWorlds®, that Hegedus had applied for and obtained in the meantime and became the main tradename for all derivative products; e.g., SimCalc MathWorlds® for the TI Graphing Calculator and SimCalc MathWorlds® for the Computer (see Fig. 6).

Since 2008, the SimCalc development team has completed a networked version of the computer software, as the team saw increasing opportunities in developing countries for using smaller computers (e.g., Netbooks) in wireless networks. As such development solidified and froze its core functionality, development was focused on making the application as usable as possible by a wide variety of users in conjunction with the SimCalc curriculum materials in the future without any more external support for development. The focus, as it had been from the start, was on creating access to important mathematical ideas to many students in simple ways.

Many decisions were made in cutting features that were not deemed functional or too confusing. For example, it was decided there was no mathematical reason to constrain the use of negative time. Classroom management tools were made simpler, and network robustness improved so now over 60 SimCalc users could connect and work together—which was first trialed in Mexico—see Fig. 7.

Fig. 7 SimCalc trial in Mexico, using over 60 connected users on a single network activity

5.1.1 Classroom Management

What was central to such development in the past few years was a deep focus on enhancing learning and motivation through mathematically meaningful participation—for that management of representational affordances was critical both in the intentional design of the activities and in the enacted curriculum in the classroom.

One affordance of a connected classroom was an explosion of student data for teachers to manage during a SimCalc activity. For example, a teacher might have 25 students in her classroom who are each working on 1 or 2 functions. These functions can have multiple forms of representation. For example, the function could be animated in the world, in addition to having a graphical (position or velocity graph) and tabular representation. Therefore, given these multiple representations for each student function, within seconds one teacher might have hundreds of mathematical representations to manage—preferably in meaningful ways. The classroom management window (see Fig. 8) was developed over several years to combat such a challenge in simple and effective ways, keeping in mind that a teacher has limited time to click through a software interface during class. At the same time, the SimCalc team believed that mathematical structure should be an emergent phenomenon and the potential of the social activity space was evolving too. Therefore, it was necessary to create a system so that teachers could progressively show the work of multiple students at once in mathematically meaningful ways.

For example, the teacher could demonstrate how groups of students differed from one another in a systematic way by hiding and showing particular students' representations. Group 1 might have been using their group-number to construct an actor whose velocity was three times their group number (see Sect. 5.2 for more informa-

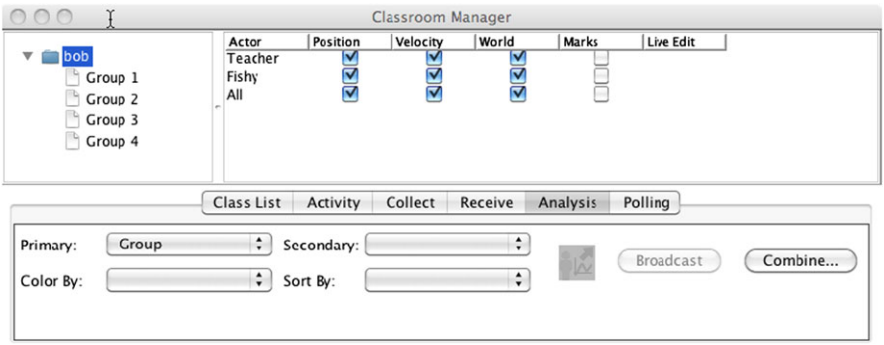


Fig. 8 The Classroom Manager interface

tion on group number). On aggregate, each group systematically varied in terms of the velocity of their actors, or the slope of their linear position graphs, e.g., $y = 3Gx$, where G is their group number. The classroom management window allowed a teacher to show or hide individual or whole group work at a click of the mouse, and progressively show other representations using a similar method, thus fusing meaning across representations. This advance in the software functionality on the desktop computer was critically important as it co-evolved with the development of the curriculum. Embedded in such curriculum were suggestions to teachers of suitable questions that proved useful in prior work for eliciting responses and building consensus within the classroom at a whole group level. The classroom management window not only allowed the teacher to do traditional network activities such as initialization and sending/collecting work, but allowed teachers the possibility for drawing attention to particular student work in progressively meaningful ways that cohered with the intentional design embedded in the curriculum.

5.2 Connectivity-Based Activity Structures

SimCalc was initially developed as a 1:1 student environment, but infusing connectivity with the dynamic, interactive features of the SimCalc learning environment created a classroom and activity space that allowed new, mathematically meaningful ways of participation for students in a mathematics classroom. From this infusion, new classroom activities exploited connectivity by designing mathematical tasks that used the natural classroom set-up of groups to offer variation in the graphs students were creating. The teacher would then aggregate the student work and choose to show all or some of their work. What followed were mathematically intense interactions among the students, and public debate within the classroom about what they saw, or where they were, in the group structure.

Emerging from this new form of student participation was an activity structure that adapted students' unique network login identifiers, which were embedded with

a student's assigned group number and count-off number within the group, into a parameter within the mathematical task at hand. For example, a student might have been asked to construct a motion where their velocity was equal to their group, and their count-off number was the starting position. Thus, the students began to create linear motions, either graphically or algebraically, which were personally meaningful to them since they incorporated their login information. Teachers also used these identifiers to aggregate student contributions into a mathematically meaningful object, e.g., a family of functions. This activity structure allows for a move from a personal individual construction to a significant group structure to a class aggregation.

5.3 Innovative Tools for Formative Assessment

The SimCalc connected classroom not only allows for private, student work to be elevated to a whole-class discussion, but there is functionality that could focus students on analyzing data in the public space (i.e., teacher's display). The polling interface allows for a teacher to pose a question to the entire class that is focused on one of the linked, Cartesian representations within the SimCalc environment (i.e., a graph or the simulated world) and, from their personal device, students specify a point in the representation that they perceive as a possible solution to the question.

The teacher could pose the question to the class, "Can you show me in the simulated world at what point does Kevin's position graph intersect Jenny's position graph?" This type of questioning involves varied student solutions, and in answering the questions, students must make sense of mathematical representations they themselves did not construct. By definition of the activity, there are at least two appropriate answers to the question, because the graphs should intersect at the beginning of the simulation and at the end. However, more answers could be possible if the graphs intersect during an activity such as Sack Race. In this case, students would have to analyze the two graphs for points of intersections and then translate those points to an appropriate point in the simulated world.

6 Dissemination of SimCalc MathWorlds®

When Hegedus initiated a commercialization trajectory for SimCalc in 2006 it was the first step towards an at-scale diffusion of the software and materials. Prior to this, disseminating SimCalc for independent implementation (i.e., without influence from the SimCalc team) was limited by the software environments lack of configurability by those outside the SimCalc team. However, as part of Hegedus' commercialization effort, the SimCalc software development team created a robust authoring environment, and added the ability for it to be implemented in four languages. These additions to the software environment have allowed SimCalc to now

be independently adapted and implemented in a number of countries, including research projects in Mexico and Greece. In order to allow true internationalization for the purpose of wider dissemination, the SimCalc team redesigned the localization support within the software, making it possible to "plug in" additional languages. Working together with some SimCalc personnel, new target language resource files could be created and plugged in, with greatly reduced technical intervention. This new ability allowed for the multilingual SimCalc software to extend beyond the development lifecycle.

7 Future Perspective

SimCalc software development existed in an environment of many influences. Cross-disciplinary tensions among software developers, researchers, and activity developers sparked discussion and innovation, sometimes in the smallest of details. However, a consistent focus on the mathematical structure guided the decisions underlying the representational infrastructure, the need for configurability to produce generative activity structures, and new forms of participation driven by a communicational infrastructure that formats the classroom according to the mathematical intentions of activities.

The activity structures that emerge out of the communicational and configuration affordances of the software extend the life of SimCalc beyond its development cycle. Development of SimCalc was finalized recently, making it important to establish a version that was stable and sustainable to be used by existing users and a wider variety of students and teachers around the world in the foreseeable future. The final authoring system allows teachers to strip away any part of the menu system to minimize the actions available to a student, enabling a more focused activity system and simpler interface.

It was also important for us to establish the software to be used in multiple languages and to date it is fully functional in English, French, Spanish, Portuguese, and Greek. While the software is commercially available, the curriculum that was simultaneously developed over a similar amount of time is freely available from the Kaput Center and can be adapted and distributed under a Creative Commons license. It was imperative for the dissemination of a learning environment that was made possible by several federal grants over 18 years, to be available, usable and configurable by many more researchers, teachers and students in the future.

References

Brown, A. L., & Campione, J. C. (1996). Psychological theory and the design of innovative learning environments: on procedures, principles and systems. In L. Schauble & R. Glaser (Eds.), *Innovations in learning: new environments for education* (pp. 289–325). Hillsdale: Erlbaum.

Dalton, S., Hegedus, S., Brookstein, A., Tapper, J., & Moniz, R. (2011). *Measuring learning in SimCalc Algebra 1 classrooms*. Technical report #3, Fairhaven: Kaput Center for Research and Innovation in STEM Education, University of Massachusetts Dartmouth.

Hegedus, S., & Moreno-Armella, L. (2010). Accommodating the instrumental genesis framework within dynamic technological environments. *For the Learning of Mathematics*, *30*(1), 26–31.

Hegedus, S., & Penuel, W. (2008). Studying new forms of participation and identity in mathematics classrooms with integrated communication and representational infrastructures. *Educational Studies in Mathematics*, *68*(2), 171–183.

Kaput, J. (1994). Democratizing access to calculus: new routes to old roots. In A. Schoenfeld (Ed.), *Mathematical thinking and problem-solving* (pp. 77–156). Hillsdale: Erlbaum.

Olive, J., & Lobato, J. (2008). The learning of rational number concepts using technology. In M. K. Heid & G. W. Blume (Eds.), *Research on technology and the teaching and learning of mathematics: research Syntheses* (Vol. 1, pp. 1–53). Charlotte: Information Age Publishing.

Roschelle, J., Kaput, J., & DeLaura, R. (1996). Scriptable applications: implementing open architectures in learning technology. In P. Carlson & F. Makedon (Eds.), *Proceedings of Ed-Media 96: world conference on educational multi-media and hypermedia* (pp. 599–604). Charlottesville: American Association of Computers in Education.