Dirk Stahlecker
6.UAP Project Report
December 15th, 2015
Supervisor: Dr. Kimberle Koile

A User-Customizable Command Ribbon For Classroom Learning Partner

Abstract

This paper describes new additions to Classroom Learning Partner software, a tablet-based classroom interaction system developed by Dr. Kimberle Koile and her research group at MIT. The additions are designed to improve a teacher's ability to specialize lessons by specifying which digital tools are available for the students to use. The work allows a teacher to choose which tools are available for each lesson rather than either having all tools available or requiring that the software be recompiled to limit the tool set.

1. Background

Classroom Learning Partner (CLP) is a networked, tablet-based application that provides students with tools to help them create, manipulate, and annotate visual representations, then share the representations with their teacher [1-3]. It also provides teachers will tools to structure and carry out lessons aimed at using visual representations to deepen conceptual understanding. CLP has most recently been used in the NSF-funded *INK-12: Teaching and Learning Using Interactive Ink Inscriptions* project (ink-12.mit.edu), which focuses on developing and testing in classrooms a set of pen-based digital tools aimed at improving upper elementary math education. The tools developed to date focus on helping students learn multiplication and division. These tools include arrays, bins, number lines, and a drawing tool we call a stamp, which facilitates creation of multiple identical images—a common task in creating visual representations in upper elementary math. Teachers create lessons, organized on “pages” in an “electronic notebook” that students open on their tablets. Figure 1 shows a screenshot of a student’s notebook opened via CLP.

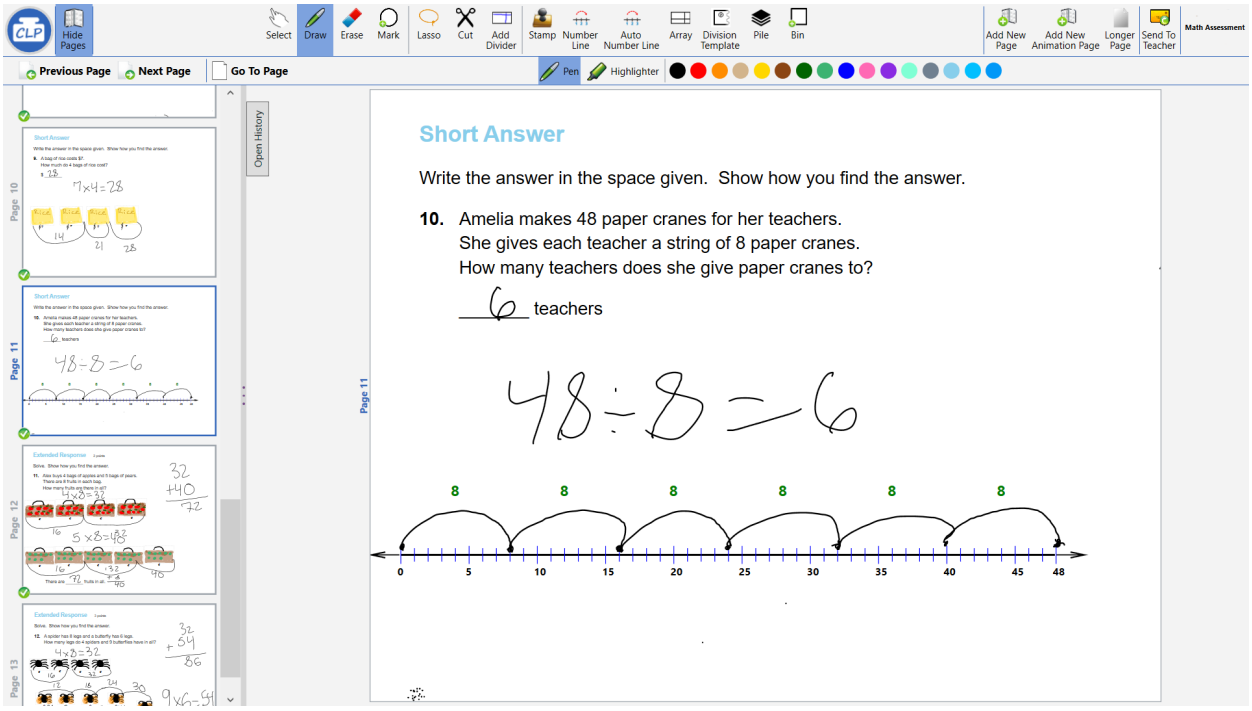

Figure 1. CLP student interface; the notebook "pages" are on the left; the current page is in the main window; commands for creating representations are shown in the top command ribbon

When working through the notebook pages, students can choose from the variety of digital tools available in order to help them solve the lesson's problems. This arrangement allows students freedom to explore different representations and decide which makes sense for them. Several different examples of student work for a multiplication problem are shown in Figures 2 through 5. The students, third graders in a classroom in the Boston area, were asked to solve a word problem that required them to multiply 5 times 8 and 4 times 8, and add the results to get 72. The examples illustrate the variety of representations that students can create using CLP's tools.

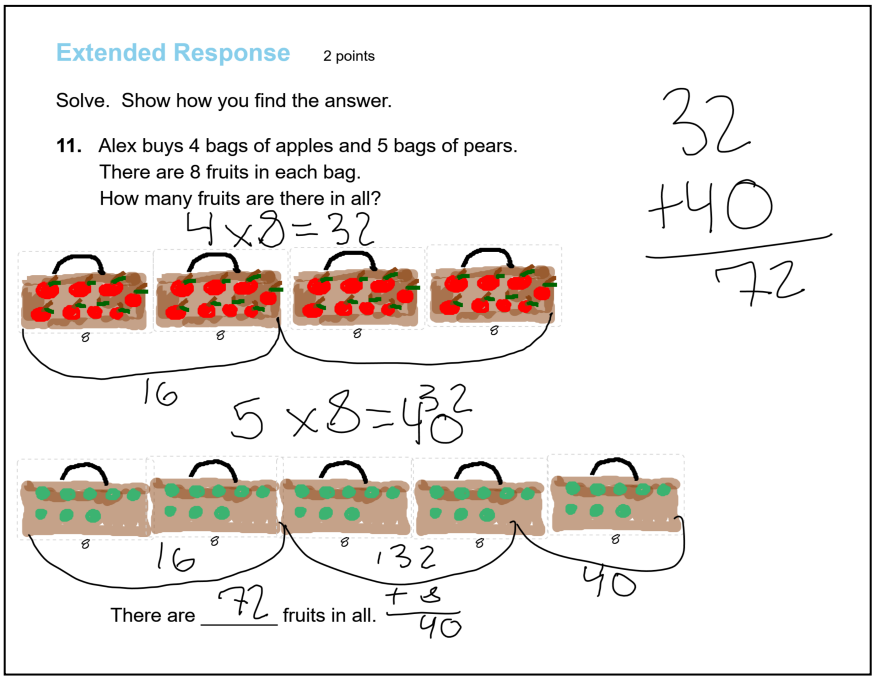

Figure 2. Stamp tool: This student solved the problem by using the stamp tool. He created a stamp for each problem containing a bag with the correct number of fruit, stamped it once per bag in the problem, and counted up the total amount of fruit in the bags.

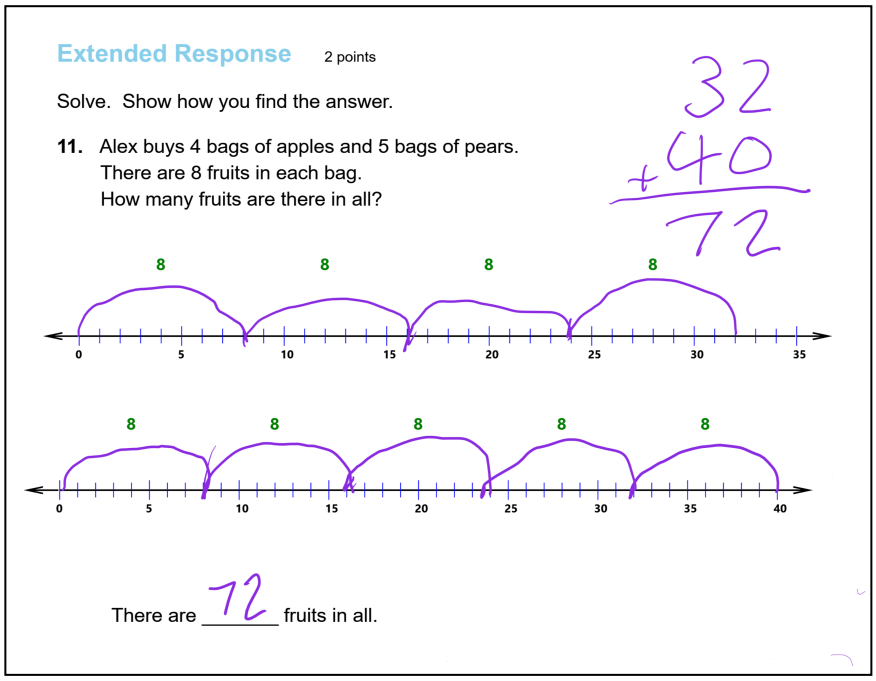

Figure 3. Number line tool: This student used the number line tool to create a number line and mark off increments corresponding to the number of fruit in each bag. The number of jumps represents the number of bags.

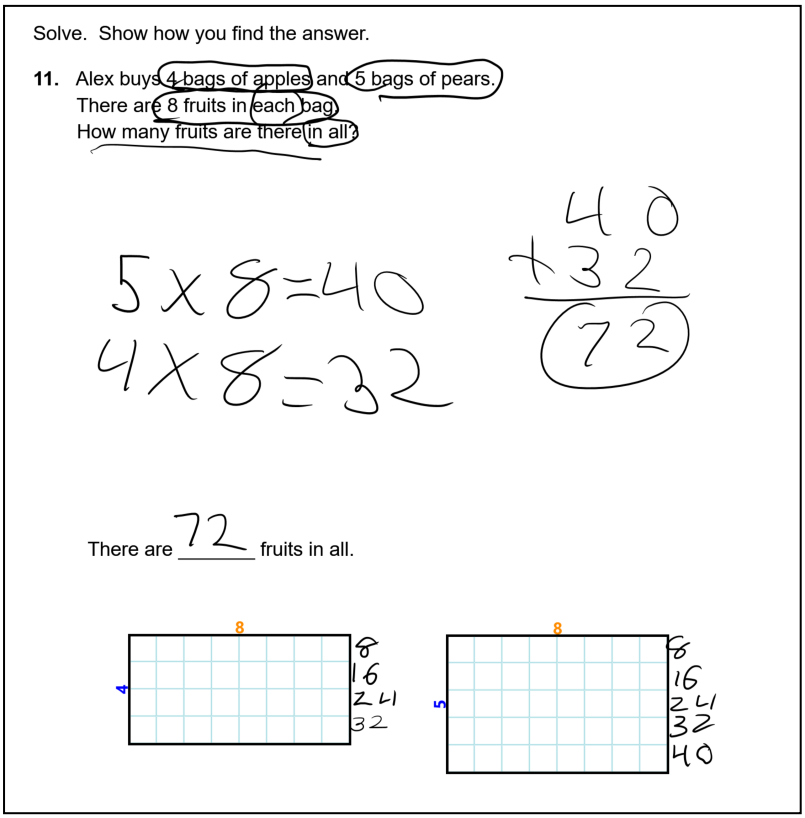

Figure 4: Array tool: This student created an array representing the two parts of the problem and added the array products together to get the answer.

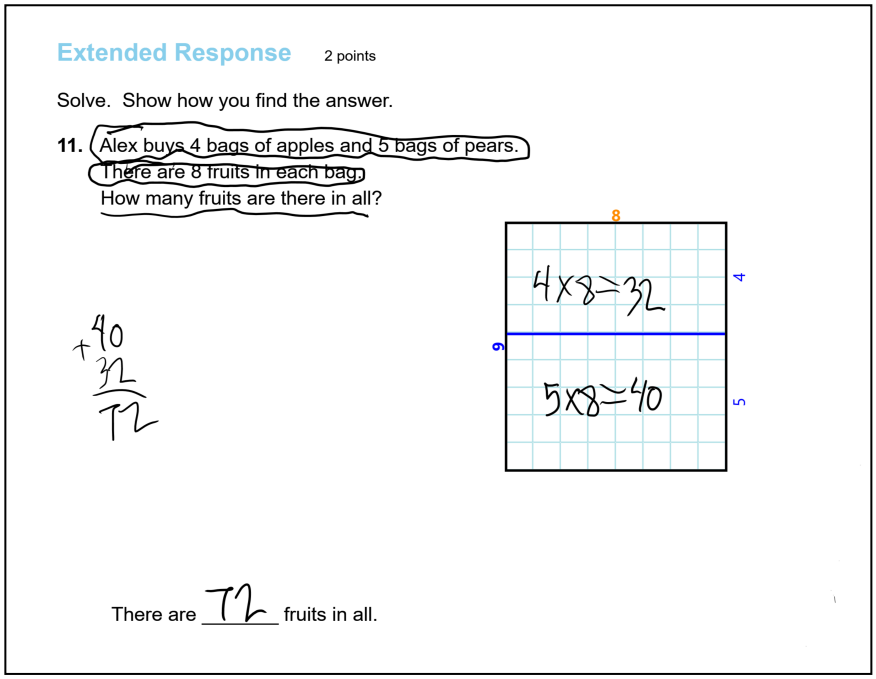

Figure 5: Divided array tool: This student combined the information given in the problem to create a single array, with a divider indicating which section correspond sto apples and which corresponds to pears.

Students electronically submit their pages to the teacher when finished with a problem. When a teacher receives the page from a student, he or she is able to view both the final representations and the process a student went through to solve the problem. The process is visible by replaying a history of the student's interactions with the tools. Machine analysis routines currently under development will also provide a teacher with information about a student's representations and process, providing a way for teachers to gain insight into what a student was thinking while solving the problem.

2. Problem

CLP contains a large number of possible tools for students to use to create visual representations when working on a problem. In the multiplication problem shown in Figures 2 through 5 students used stamps, number lines, and arrays, stamps. There are other tools available as well that were not appropriate for this particular lesson, e.g., a tool called a division template, which is aimed at helping older students deepen their understanding of division [2]. Not all lessons necessitate use of the same tools, and a teacher may wish to limit the toolset available for a particular lesson. In the current implementation of CLP, limiting the set of tools available for student use requires recompiling the software so that only the desired set of tools is visible on the top command ribbon. This project provides a way to let teachers easily specify which tools are available for a particular notebook, eliminating the need to recompile the software for use with a particular notebook.

3. Solution

This project gives teachers the option of selectively enabling and disabling the tools available for students to use. This control could potentially take place at three different levels: the user level, notebook level, and the page level. For this project, we decided to focus on the notebook level because each notebook typically contains a single curriculum unit; the same tools are often going to be appropriate throughout a particular unit. Changing tools at the page level could be useful, but also could lead to potentially confusing situations for the students if the students expect a certain set of tools to be present. Preferences stored at the user level would typically be related to the particular software environment settings, such as visual settings, layouts, or other customizable options. This preference setting is a useful feature, but not appropriate for a lesson set up by a teacher; the teacher will know what the learning goals are for a lesson and which tools are most appropriate.

The first step was determining how to store the toolset preferences. Particular care was taken to ensure a storage method that fit well into the design pattern that the application embodies, namely the Model View View-Model pattern [4]. Specifically it was necessary to consider the behavior of saved files in relation to the networking architecture of CLP: In the classroom, the teacher opens her notebook, which has been created using a CLP authoring tool, and the notebook is distributed to each student when he or she logs in to a student machine. The notebook also is displayed on a machine connected to a projector. The preferences stored by the teacher in the creation of the notebook must be stored with the notebook or transmitted across the network with the notebook to student and projector machines. We decided to use XML to save the preferences, as it fit the design pattern, is easily edited, and is human-readable. Inside the file save structure of CLP, we added a new preferences XML file that is referenced by the teacher notebook. Student notebooks will contain the same reference and simply require that the XML file be transmitted over the network when the notebook is loaded onto the student machines.

Once the preferences were created in the code, serializing them to XML was straightforward. The creation of the preferences, however, caused some difficulties because a new structure was needed in order to interact dynamically with the individual buttons. Previously, no dynamic control with possible for individual buttons, and changing the layout and/or showing or hiding buttons required manually changing the code and a creating a new software build. In order to solve this problem, we implemented a new low-level class that expanded the abilities of the view model to manipulate the view with specific reference to the tool buttons. This view model now derives data from the XML preferences file.

This solution is very simple overall: Because all the preferences are stored in three XML files in a known location, these files can easily be transferred over the network whenever the teacher pushes updates to the students' tablets. There is no need for preference updates to be pushed more frequently than this, as they normally happen when creating or editing a notebook.

This new functionality allows for the basic behavior that we desired: showing and hiding buttons based on saved preferences. Next up was determining how the user, typically the teacher, would interact with this new functionality. Several considerations were kept in mind when designing the user interface. Foremost was simplicity and ease-of-use. Teachers already spend large amounts of time working on the lesson plans, and if CLP were to cause a significant burden on their effort or time, it would likely not be used. Any new feature must be simple enough that teachers feel comfortable with it; it should be understandable virtually at a glance.

We chose to create a new dialogue window to house the toolset selection feature. The only existing dialogue for options and settings in CLP was for such operations as creating and saving a notebook or setting user preferences, and it was accessible via a different full-screen window and a set of nested submenus. We chose to have the new dialogue window overlay the main

CLP screen, so that users could be reminded of the context in which the tools would be used. It was necessary to position the new dialogue window and access button such that it was clear to the user that the preferences being sent were to be applied to the notebook as a whole. In order to make this distinction clear, we placed the button in the upper right corner, in alignment with the buttons that this menu would allow us to manipulate. The placement of this menu was also designed to accommodate future expansions and features targeted at modifying properties of the notebook, similar to what this project has done. Rather than adding a button to the already crowded top ribbon bar, we can leverage an existing feature by simply making the information box in the upper right clickable. This option both saves space and also makes intuitive sense: information about the notebook is already displayed here, so piggybacking on top of it makes sense.

We decided to keep the user interface for the preferences modal itself as simple as possible. Each button only has two states, visible or not visible, therefore a simple checkbox is sufficient. However, there is a nuance in determining visibility. There may be a case in which the teacher wishes to hide a button for his or her student but not hide a button on the teacher's tablet. It is also possible that the teacher may wish for a button to be visible on the projector tablet but not the students' tablets, or vice versa. Therefore, we built in the ability to determine visibility for teacher, student, and projector independently. The preference dialogue window is shown in Figure 6.

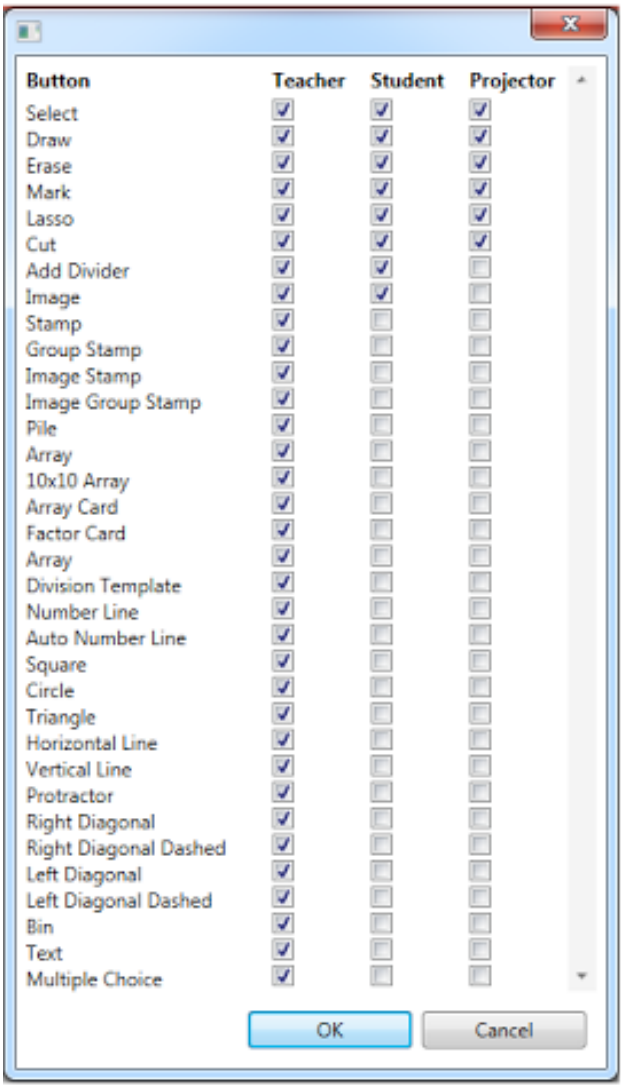

Figure 6: Preferences UI window

4. Future work and implications

The work on this project is useful for both teachers and CLP developers. Previously, it was necessary to compile a new build every time the button layout was modified. Besides being time-consuming and tedious to do, this requirement led to a large number of virtually identical but slightly different software versions in production in different places. Now, utilizing the new feature added, this compilation is no longer necessary. Visible buttons and CLP software version number are no longer intricately linked and can be modified independent of one another. In addition, the ability to customize CLP for particular teachers or classrooms is very useful; customization can help the teacher and students learn to use CLP much more quickly by eliminating buttons not needed by a particular lesson. Along the same lines, it is now possible

for specialized buttons to be created for particular curricula that may not be useful or necessary for every curriculum, further increasing customizability.

The implications for teachers are also worthy of note. Teachers now have significantly greater ability to tailor lessons with much finer detail, as they can specify which tools are available for students, as has been discussed in previous sections in this paper. In the future, we would like to test expanding the preference functionality to allow for tools to be customized at not just the notebook level, but also at the page level. This added functionality would allow teachers to specify which tools are available on a per-problem basis, allowing the controlled introduction of tools and specialized problems to practice use of the tools. As noted earlier, however, students may be disconcerted by finding that the tools change with a page: When allowing available tools to be specified at the page level, there is a potential for a seemingly inconsistent or scattered UI. Care would have to be taken to ensure that the showing and hiding of buttons while flipping through pages is not a visual distraction that makes the application look jumpy. One possible solution is to grey out the buttons to indicate they are unavailable, rather than hiding them. Classroom testing is needed to determine if these ideas are worth pursuing.

Each new feature makes CLP potentially more cumbersome to use. Effort must be made to keep the user interface very intuitive and simple, especially on the student side, so as to not take away from instructional time or interrupt students' thinking about the math. The new functionality described in this report is consistent with this goal and provides much-needed customization that will help make CLP even more useful in classrooms in the future.

References

1. Koile, K. & Rubin, A. (2015) Machine interpretation of students' hand-drawn mathematical representations. In *Impact of pen and touch technology on education*. Hammond, T., Valentine, S., Adler, A., and Payton, M. (Eds.) New York: Springer.
2. Koile, K. & Rubin, A. (2015a) Tablet-based technology to support students' understanding of division. In *Impact of pen and touch technology on education*. Hammond, T., Valentine, S., Adler, A., and Payton, M. (Eds.) New York: Springer.
3. Rubin, A, Storeygard, J., & Koile, K. (2015) Supporting special needs students in drawing mathematical representations. In *Impact of pen and touch technology on education*. Hammond, T., Valentine, S., Adler, A., and Payton, M. (Eds.) New York: Springer.
4. Smith, Josh. (2009) Patterns - WPF Apps With The Model-View-ViewModel Design Pattern. In *MSDM Magazine*.