Class.ly: Helping Students Collaborate

Timing-Based Analysis of Common User Actions Across Collaboration Tools

Mitchell J. Masia
Computer Engineering
Vanderbilt University
Nashville, TN, 37235, United States of America
mitchell.j.masia@vanderbilt.edu

Parker A. Klein
Computer Science
Vanderbilt University
Nashville, TN, 37235, United States of America
parker.a.klein@vanderbilt.edu

***Abstract*— Class.ly is a tool built to integrate with the Vanderbilt Univeristy Your Enrollment System (YES) in order to increase collaboration among Vanderbilt undergraduate students. Class.ly's main objects are to optimize common cases of collaboration: messaging and scheduling. The creators of Class.ly performed a time-based comparative analysis of these actions with alternative collaboration systems Blackboard and Piazza. The study also encompassed learning-curve determination while using the Class.ly application to observe temporal and memory strain of application users, and how repetition can help potential users become increasingly quick in performing the scheduling task. Class.ly was extremely competitive in the time-domain again the more established systems, due to its intuitive interface and common case optimizations.**

Keywords—collaboration, class.ly, blackboard, piazza, undergraduate, vanderbilt, automation, messaging, scheduling.

I. Introduction

The rise of mobile technology has led to an exceptional increase in collaboration in industrial, social, and academic environments. Despite this explosion of communication, acquiring information to facilitate contact has remained a static process. Individuals must ask for contact information and spend minutes typing this information into various phones, before collaboration can commence. This is an especially tedious process for Vanderbilt University undergraduate students. VU Students likely have one or more group projects in several of their concurrent classes, each of which will require a contact information swap. This process is not only tedious, but can be very frustrating when mistakes are made. The static contact exchange does not facilitate open communication lines, a core value of our increasingly connected society, nor does it facilitate transparency- a key value among working groups.

Being Vanderbilt Undergraduates themselves, the creators of Class.ly have faced the annoyance of information swapping for far too long. This led to the development of the Class.ly application. The goal of Class.ly is to make two of the most basic collaboration tasks, messaging, and scheduling incredibly simple. Class.ly excels in its utility by automatically generating chatrooms via enrollment information, and allowing groups to schedule working sessions with only a few keystrokes. Existing tools such as Blackboard and Piazza implement far more than these two features; however, Class.ly follows the paradigm “optimize for the common case”.

Class.ly simplifies messaging and scheduling processes by integrating with the Vanderbilt University Your Enrollement System (YES). Upon a Vanderbilt undergraduate signing up for Class.ly, the system will automatically pull their enrollment information from YES. Using this data, Class.ly automatically creates a “Course,” a representation of a Vanderbilt University course. Each Class.ly Course incorporates a custom group creator, message board, and a meetup scheduler. Integrating with YES eliminates the need to swap contact information as necessary group-based communication can occur in-app.

Without spending time on sharing contact information, undergraduates can focus more on what matters: accomplishing their work. In an age of automation, where user experience supposedly trumps well-implemented functionality, it is amazing that students and professionals are still faced with the time-consuming problem of contact management. Class.ly aims to make the collaboration process a lean one by cutting out time and effort-based waste. With Class.ly's proposed functionality, students will be able to work smarter, not harder.

II. System Description

A. Originating Requirements

- The system will seed pertinent user information upon sign-up.
- The system will automatically create chatrooms.
- The system will allow meeting scheduling.
- The system should suggest best-case meeting times for group members.
- The system will allow custom group creation.
- The system shall allow opt-out options.

B. System-Level Requirements

- The system should integrate with Vanderbilt YES to pull user enrollment data.
- The system should implement a familiar instant-message based interface for messaging.
- The system will not limit message length.
- The system will allow users to manually create meetups.
- The system should integrate with 3rd-party calendars to find preferred meeting times for group members.

- The system will allow users to mute messages for specific courses and groups.
- The system should allow users to opt out of a specific class or group.
- The system should allow users to delete their profiles.

C. Component-Level Requirements

- The system will implement event-based functionality.
- The system will have limited CRUD access to YES resources via the internal Vanderbilt YES REST API.
- The system should integrate WebSocket functionality to alert users of new messages in specific chatrooms.
- The system should integrate with Google Calendars.
- The system should integrate with Apple iCal Calendars.

D. Minimum Viable Product (MVP) Functionality

Due to time and development effort constraints, only a specific subset of features outlined in the Class.ly proposal and requirements will be put into the final production version of the application for Human-Computer Interaction demonstrations. To maintain key principles guiding the Class.ly creators, these features will be the custom group creation, limited messaging capabilities, limited meetup scheduling capabilities, and opt-out functionality.

Once a user has navigated to a specific course, they will be able to create new groups. A group is a specific collection of users all enrolled in a particular course. That is the only commonality shared by the group members. Once a group is created and members are added, the group encapsulates its own chatroom and available meetup scheduler. Messages and meetups created within this group only persist in the context of the group for future references.

The specific chatroom within a group implements the commonly used instant-message type interface mimicking closely the “GroupMe” messaging interface [1]. Upon entering the messaging interface, users are able to send arbitrary length, arbitrary character messages using UTF-8 encoding. In the MVP, used to demonstrate functionality, messages are temporarily stored locally on the client device; however, they are not persisted in the Class.ly database.

Upon entering the meetup scheduling feature in the Class.ly MVP, users are able to create arbitrary name, arbitrary time meetups and invite specific users in the current group to attend. Similar to the messages, these meetups persist in local storage, but are not reliably stored in the database to be accessed at a later date or connection session.

From initial conversations with potential users about necessary functionality, the Class.ly team has implemented one extraneous, but highly important feature: the ability to mute or “opt out” of a specific group or course. In initial discussions with users, the team found that a major worry was that of annoyance. Given that the average Class.ly user has four courses, each of which have two or more group projects, the notifications of messages and meetups could easily become overwhelming. This prompted the fabrication and implementation of the opt-out feature that allows a user to essentially disable specified notifications. This feature is fully implemented in the Class.ly MVP.

In theory, using the event-based responsive system architecture, users would receive PUSH notifications based on events concerning them (being added to a group, receiving messages, being invited to a meetup, etc.). Unfortunately, due to time constraints, this functionality is not implemented in the MVP. Far more time in this iteration has been spent on interface design and implementation to make the system simple to use and optimized above alternatives for the most common problems faced by the target users.

E. Development Methods

Class.ly is built on the Ionic Framework [2] which utilizes the “Write Once, Run Everywhere” (WORE) paradigm. The WORE paradigm is the idea that an application developer should be able to develop one code-base and run it on various platforms. In the case of Class.ly, the idea is to build a web application that can run in a browser context, Android Operating System on a mobile device, or on iOS on an Apple mobile device. The WORE paradigm has long been criticized for the lack of nativity, forced feel of interactions, and poor application performance over various platforms. Ionic, built upon Apache Cordova [3], mitigates these concerns by exposing the native hardware of these platforms via virtualization techniques to make them available to application developers. This means that application developers are able to access components such as the GPU for advanced image processing or rendering in real-time, achieving a high level (almost native) of performance. Upon completion of the application, Ionic allows developers to package and deploy web applications that run natively on specific Android and iOS versions.

Ionic Framework supports the development of web applications using the Model-View-Controller (MVC) pattern on the front end. MVC insists that users create data models and specific views to display that data. The models and views are then wired together via the controllers which act as a transport and manipulation layer for storing and displaying data from various models [4]. Ionic insists that application developers use the MongoDB-ExpressJS-AngularJS-NodeJS (MEAN) stack for front and back end development. This means that applications are written entirely in the Node JavaScript language with MVC on the front end and an exposed RESTful API (via ExpressJS) on the back.

Class.ly is developed using the MVC pattern and the Node JavaScript language. The application is backed using the NoSQL document-based data store MongoDB [5]. As Class.ly does not yet integrate with Vanderbilt YES, the MongoDB store contains mock data on courses, users, messages, and meetups to seed the demonstration Class.ly application. For more information on pertinent data models, configuration, and Ionic Framework packages, please see the Class.ly code base hosted publicly on GitHub at https://github.com/masiamj/CS3892-HCI-Class.ly/tree/master/Class.ly.

F. *Interface Design*

One of the primary goals of Class.ly is to be mobile. Such a large percentage of communication nowadays happens on-the-go. Hyperengaged Vanderbilt University undergraduate students are often running from meeting to meeting. Class.ly is designed to account for this normalcy with incredibly simple interfaces for messaging and scheduling. These interfaces are primarily designed for mobile, and as such, the browser-based user experience suffers slightly. Class.ly was designed with the pain-points of alternatives such as Blackboard and Piazza in mind. Class.ly is designed to reduce temporal and memory load on the user, exposing familiar components to systems undergraduates use everyday. Class.ly primarily draws inspiration from the application "GroupMe" which has a clean and intuitive instant-messaging interface.

Class.ly uses both side-drawer and tabbed-navigational menus. Similar to commonly used application like Facebook or Venmo, Class.ly uses its side-drawer for configuration and setting information. The main functionality exists in tabs, located at the bottom of the screen on mobile and web-browser contexts. Class.ly uses the prototypical "proactive" blue theme for Android and iOS navigation and informational (top) bars.

III. User Testing Methods

Multifaceted user testing of Class.ly was conducted in order to test several parameters: usefulness of the application, collaboration effectiveness against Blackboard, collaboration effectiveness against Piazza, learning percentage calculation, and user adoption predictions.

Qualitative analysis was performed to determine usefulness of the application as well as important future functionality to be implemented. Quantitative analysis was performed on the collaboration effectiveness against both Blackboard and Piazza. The effectiveness is highly important and determined via timing-based comparison. The reason for this is the necessity for on-the-go communication and the inherent speed required. Quantitative analysis is also implemented in finding the learning percentage calculation of users over time while using Class.ly to perform a specific action.

A. *Abbreviations and Acronyms*

In the course of user testing, it is important to note several abbreviations used in logging time-series data about user performance across the Blackboard, Piazza, and Class.ly systems. Necessary abbreviations include:

- **TMB –** Time to Message on Blackboard
- **ATMB –** Average Time to Message on Blackboard
- **TMP –** Time to Message on Piazza
- **ATMP –** Average Time to Messgae on Piazza
- **TMC –** Time to Message on Class.ly
- **ATMC –** Average Time to Message on Class.ly
- **TM1C –** Time to Create Meetup #1 on Class.ly
- **ATM1C –** Average Time to Create Meetup #1 on Class.ly
- **TM2C –** Time to Create Meetup #2 on Class.ly
- **ATM2C –** Average Time to Create Meetup #2 on Class.ly
- **TM3C –** Time to Create Meetup #3 on Class.ly
- **ATM3C –** Average Time to Create Meetup #3 on Class.ly

B. *User Testing Participants*

User testing for Class.ly consisted of 10 combined interview and observation periods each lasting approximately 15 minutes. To begin the interaction with each of the 10 users, each participant filled out and introduction survey to gather demographic information about the test group. As reflected by the demographics, it is important to note that the key diversifying characteristic among participants is not gender or age, but academic field of study. This is chosen purposely to acquire a broad range of individuals who inherently have different levels of use of Blackboard and Piazza due to their differing departments and backgrounds. From the introduction survey, several pieces of important information was gathered.

- 80% of the participants are male
- The mean age of the participants is 21.25 years
- The median age of the participants is 21 years
- The mode age of the participants is 21 years
- Every participant was of senior status at Vanderbilt University
- Participants' academic majors include: Economics, HOD, Mechanical Engineering, Computer Science, Math, Art, and Biomedical Engineering
- Participants' academic minors include: Spanish, Financial Economics, Materials Science, and Engineering Management
- Average Experience on both Mobile and Desktop devices is between "High" and "Expert"
- Average Experience on Blackboard is between "Medium" and "High"
- Average Experience on Piazza is between "None" and "Low"
- The only users with Piazza experience have a Computer Science academic major
- No participants had physical or mental impairments

Graphical representation of the user testing participants is shown below

Figure 1: Distribution of Blackboard and Piazza Experience

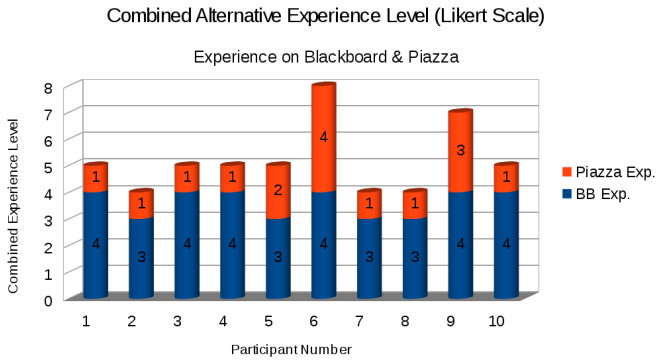
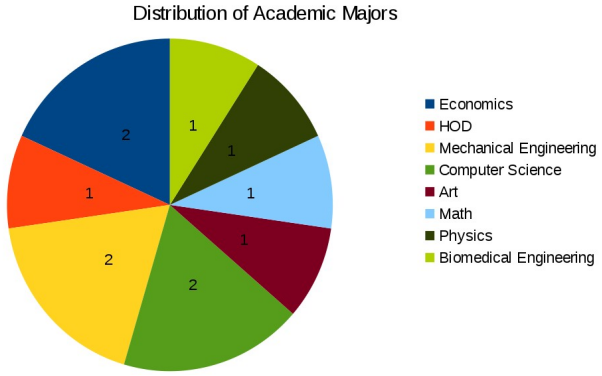


Figure 2: Distribution of Academic Majors



C. *Experimental Design*

User testing for Class.ly took place in a continuous sitting in the Featheringill Hall Atrium at Vanderbilt University. Throughout a three hour period, the creators approached idle students studying and asked for their participation in a user study. The participants then accompanied the creators to their consistent study environment to perform the testing. To begin experimentation, the users were read a script explaining the purpose of Class.ly and the testing taking place. At key moments, they were prompted for verbal responses to indicate that the user testing could proceed.

After completing the script, participants were asked to complete the introductory survey, from which the results above were collected and aggregated to learn about the testing population sample. Upon completion of the introductory survey, operational user testing scenarios began.

The first operational testing scenario was to send a message to an entire class using the Blackboard collaboration system. To begin the experiment, the facilitators asked the participant to navigate to a specific class on the Blackboard dashboard (starting from the logged-in state), and send a message to the entire class that reads “The quick brown fox jumped over the lazy dog.” Once the participant acknowledges the instructions, the facilitators tell them to begin the task and an external timer is started. As the activity takes place, facilitators observe and note (qualitative), but do not guide, the actions of the user monitoring their procedure to send the message. Upon completion of the task, the time to completion is acquired and logged for future analysis.

The second operational testing scenario was to send a message to an entire class using the Piazza collaboration system. Facilitators asked participants to navigate to a specified class (starting from a logged-in state), and send a message to the entire class with the same message: “The quick brown fox jumped over the lazy dog.” Again, upon acknowledgment, a timed test to complete this task begins.

The third operational testing scenario was to send a message to a specified class using the demonstration-version Class.ly application. Procedures to complete and monitor the activity mimic the first and second operational scenarios.

The fourth and final operational scenario is a multi-part activity. This scenario encompasses three iterations of the same activity. The activity required is that participants create a meetup- one of the core features- using the Class.ly application. Before the first iteration, users are asked to create a meetup in “FGH” at 12:00 PM on Friday, November 13, 2015 (11/13/15), as well as invite one user in their class to attend the meetup. This is a relatively simple task to complete on the Class.ly platform. This same activity is complete two more time in succession, with small breaks, in order to derive the activity learning percentage for completing this activity.

Finally, after completing all of the operational scenarios, participants are asked to complete a test exit survey asking them open-ended, qualitative questions about their potential to use the application, the most useful functionality to them, as well as desired functionality in a production version of Class.ly.

Data from the introduction survey, operational scenarios, and exit survey were collected and aggregated into a spreadsheet for further analysis. Examples of the introduction script, introduction survey, operational scenario data collection forms, and exit surveys are included in *Appendix A* of this report.

D. *Test Apparatus*

To test the Class.ly application, test participants were introduced to the browser-based usage context. Though Class.ly was not fully optimized for the desktop browser, but for mobile usage, for consistency (some users may not have Android/iOS experience) and speed of serving, the creators decided to run tests using a desktop computer. Testing occurred on a MacBook Pro with no external accessories. Participants did not receive any specific training on Blackboard and Piazza on purpose (to simulate a new user), and they only received a brief introduction to the purpose served by Class.ly.

The commonality between the tests on Blackboard, Piazza, and Class.ly was that all three alternative systems started from the logged in state. This experimental design choice was made because each system requires different credential information to log in (username vs. email), creating nonstandard user testing conditions.

IV. Experimental Results

The experiments described in the sections above are designed to test several key hypotheses. The hypotheses cover qualitative and quantitative bases such as the usefulness of Class.ly, user adoption projections, and intuitiveness of the

interface (based on time to complete a task). The specific hypotheses explained in this paper include:

1. 70% or greater of user testing participants would use a production version of Class.ly.
2. Average time on task of sending a message to an entire class on Class.ly is shorter than on Blackboard or Piazza.
3. Average time on task of sending a message to an entire class on Class.ly is shorter (more intutive) for first time Class.ly users than first time Piazza users.
4. Sending a message to an entire class on Class.ly is optimized for a more broad range of user experience than Blackboard or Piazza.
5. Users will achieve less than 50% learning percentage when creating a meetup on Class.ly.

To begin analysis, basic statistical information will be presented, followed by a more in-depth statistical decomposition and exploration of the results as they pertain to the hypotheses above. Analysis of basic timing information from the operational scenarios is as follows (please reference *Abbreviations and Acronyms* section of this report for reference):

- ATMB = 44.062 seconds with a standard deviation of 11.765 seconds
- ATMP = 37.401 seconds with a standard deviation of 11.947 seconds
- ATMC = 20.274 seconds with a standard deviation of 4.558 seconds
- ATM1C = 65.202 seconds with a standard deviation of 35.732 seconds
- ATM2C = 25.152 seconds with a standard deviation of 3.997 seconds
- ATM3C = 18.210 seconds with a standard deviation of 6.118 seconds
- 80% of participants dictated they would use a production version of Class.ly
- 100% of participants dictated that Class.ly was more intuitive than Blackboard
- 100% of participants dictated that Class.ly was more intuitive than Piazza

A. Hypothesis 1: User testing participants would use a production version of Class.ly

This question was asked under the binary "yes" or "no" paradigm, forcing users into a category. This qualitative answer transforms to quantitative data, following a Bernoulli distribution. The probability mass function (pmf) of the Bernoulli distribution is

$$q=(1-p)\ for\ k=0,\ p\ for\ k=1$$

Because 8/10 users said they would use a production version of classly, p = 0.8. The variance of this distribution is represented by

$$\sigma^2=p(1-p)=pq=(0.8)(0.2)=0.16$$

When estimating the number of successes (participants that *would* use Class.ly), the Binomial distribution can be used to test the null hypothesis

$$H_0=Less\ than\ 7\ user\ test\ participants\ will\ use\ a\ production\ version\ of\ Class.ly.$$

To use this approximation, parameters are

$$\mu=np=mean\ of\ binomial\ sampling\ distribution=8$$

$$\sigma=\sqrt{\sigma^2}=standard\ deviation=1.26$$

$$n=the\ number\ of\ opportunities=10$$

$$k=the\ stipulated\ number\ of\ successes=7$$

$$z=\frac{(k-\mu)\pm 0.8}{\sigma}=\frac{(7-8)-0.8}{1.26}$$

Using a publicly available Binomial approximation tool, the p-value indicating that 7 or fewer participants would use the application is found to be 0.322. As $p > 0.05$, this result is not statistically significant, and the null hypothesis cannot be rejected. This means that it is possible only 7 or fewer participants would use a production version of Class.ly.

B. Hypothesis 2: Average time on task of sending a message to an entire class on Class.ly is shorter than on Blackboard or Piazza

The results used for this hypothesis test were collected during user testing during the first, second, and third operational scenarios. The hypotheses in question are

$$H_{01}=ATMB\le ATMC=ATMB-ATMC\le 0$$

$$H_{01}=ATMP\le ATMC=ATMP-ATMC\le 0$$

Because the population of samples taken is only 10 samples deep, hypothesis will use the difference between two population means with Student's t distribution and n_1+n_2-2 degrees of freedom. Other parameters for this test are

$$t=\frac{(\bar{x}_1-\bar{x}_2)-(\mu_1-\mu_2)_0}{\sqrt{\frac{\sigma_1^2}{n_1}+\frac{\sigma_2^2}{n_2}}}$$

$$\bar{x}_1=a\ sample\ point\ of\ Blackboard,\ Piazza$$

$$\bar{x}_2=a\ sample\ point\ of\ Class.ly$$

$$\mu_1 = mean\ of\ Blackboard\ ,\ Piazza\ message\ time$$

$$\mu_2 = mean\ of\ Class.ly\ message\ time$$

Using the information above, the results of a comparison of Blackboard vs. Class.ly time on messaging task is shown below.

$$With\ a\ 95\ percent\ CI\ ,\ p = 0.00468$$

As $p < 0.05$, the result is statistically significant, and the null hypothesis H_{01} is rejected. The total sample size to achieve this result is 3 samples from Blackboard and 3 samples from Class.ly, resulting in 6 total population samples. This clearly means that the intuitiveness of the Class.ly interface (measured by speed) is greater than that of Blackboard.

Using the same procedure, the time on task comparison is completed between Piazza and Class.ly to determine their respective intuitiveness readings. The results are as follows.

$$With\ a\ 95\ percent\ CI\ ,\ p = 0.0042$$

As $p < 0.05$, this result is also statistically significant, and the null hypothesis H_{02} is rejected. The total sample size to achieve this result is 5 samples from Piazza and 5 samples from Class.ly, resulting in 10 total population samples. This clearly means that the intuitiveness of the Class.ly interface (measured by speed) is greater than that of Piazza.

For posterity, this same population mean comparison was performed between the Blackboard and Piazza collaboration systems. Using the null hypothesis

$$H_{03} = ATMB \leq ATMP = ATMB - ATMP \leq 0$$

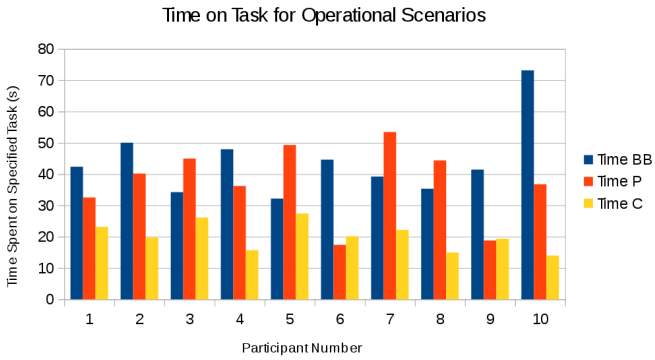
The results of this experiment are as follows

$$With\ a\ 95\ percent\ CI\ ,\ p = 0.518$$

As $p > 0.05$, this result is not statistically significant, and the null hypothesis H_{03} is not able to be rejected. This clearly means that the intuitiveness of the Blackboard interface (measured by speed) is neither greater nor less than the intuitiveness of the Piazza interface. This corresponds very well with the data collected during user testing observation, and will be discussed in the *Discussion* section of this report.

Graphical results of the time on task study for Blackboard, Piazza, and Class.ly is shown below

Figure 3: Comparative Time on Messaging Task



C. *Hypothesis 3: Average time on task of sending a message to an entire class on Class.ly is shorter (more intutive) for first time Class.ly users than first time Piazza users*

The Piazza collaboration tool is mainly used in Computer Science classes by students with the Computer Science major.

Because many of the user testing participants were not CS majors, they had little to no experience with Piazza. Similarly, these users had no experience with Class.ly. This hypothesis is solely meant to test how new users react to the Piazza and Class.ly interfaces. This comparison is another way to observe the comparative intuitiveness of both interfaces and test if Class.ly is properly optimized for the messaging activity, more so than Piazza.

The population for this study (B) consists only of users who are brand new to Piazza (and naturally to Class.ly). The same procedure from above (the difference of population means with Student's t distribution) to perform the test on hypothesis

$$H_{04} = Average\ Time\ of\ Message\ on\ Class.ly\ by\ B - Average\ Time\ of\ Message\ on\ Piazza\ by\ B \geq 0$$

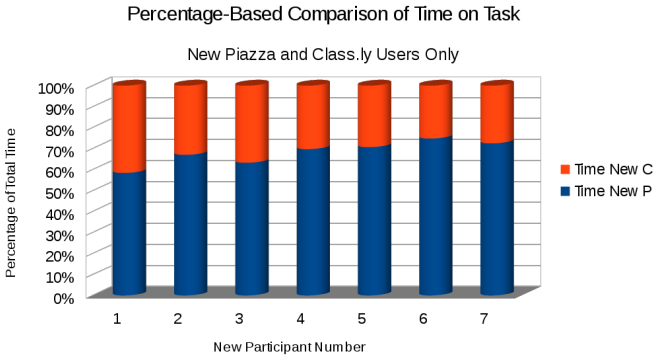
For reference, all values used in these tests are included in Appendix A of this report. Results of the hypothesis test are

$$With\ a\ 95\ percent\ CI\ ,\ p = 0.16995$$

As $p > 0.05$, this result is not statistically significant, and the null hypothesis H_{04} is not able to be rejected. This indicates that the intuitiveness of Class.ly is not decidedly greater than that of Piazza for first time users of both systems. The significance of this result will be discussed in the *Discussion* portion of this report.

A graphical representation of the comparison between first time Piazza and Class.ly users is shown below

Figure 4: Comparison of Time on Messaging Task for First Time Users



D. Hypothesis 4: Sending a message to an entire class on Class.ly is optimized for a more broad range of user experience than Blackboard or Piazza

This hypothesis can be analyzed via simple value comparison of the sample standard deviations of the time on messaging task for Blackboard, Piazza, and Class.ly. The first step in this calculation is to normalize the units of standard deviations of the three samples by dividing by the lowest (the sample standard deviation of Class.ly). This yields

$$\sigma_{Blackboard} = 2.58\ Standard\ SD\ units$$
$$\sigma_{Piazza} = 2.62\ Standard\ SD\ units$$
$$\sigma_{Class.ly} = 1\ Standard\ SD\ units$$

It is important to note here the high spread of values in the sample sets of Blackboard and Piazza, as shown by their high standardized standard deviations. This indicates that over all users with their varying degrees of experience over all three systems, the range of values in the time on messaging task study was on the Piazza system, followed by Blackboard, followed by Class.ly. This indicates that different users will not only interpret the interface in different ways; however, their lasting impression and the "stickiness" of the application will vary across user segments for Blackboard and Piazza. Contrarily, Class.ly has a comparatively low spread of values, indicating that users generally take the same path (similar interpretations), and will encounter a standardized user experience when using the application. Further discussion on this result can be found in the *Discussion* portion of this document.

E. Hypothesis 5: Users will achieve less than 50% learning percentage when creating a meetup on Class.ly.

Learning percentage in this context refers to the inverse in reduction of the time needed to perform a certain task over multiple attempts. In other words, it is the percentage of the time on task for the first attempt needed for the second attempt. In mathematical form,

$$learning\ percentage = \frac{t_1}{t_0}$$
$$where\ t_1\ is\ the\ time\ necessary\ for\ attempt\ 2$$
$$where\ t_0\ is\ the\ time\ necessary\ for\ attempt\ 1$$

By claiming in the hypothesis that users will have less than a 50% learning curve, the creators of Class.ly anticipate that the time to create the second meetup (operational scenario four) will be less than 50% the time to create the initial meetup in this scenario. The null hypothesis for this test is

$$H_{05} = Learning\ percentage\ will\ be\ greater\ than\ 50\ percent$$

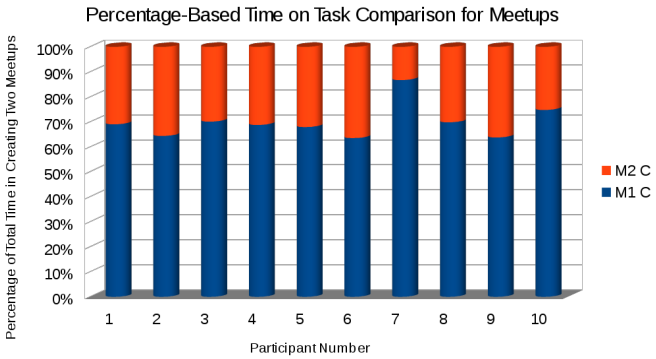
Using the equation

$$z = \frac{p - p_{exp}}{se(p)}$$
$$where\ p\ is\ the\ observed\ proportion$$
$$where\ p_{exp}\ is\ the\ null, expected\ proportion$$
$$where\ se(p)\ is\ the\ standard\ error\ of\ p$$
$$se(p) = \sqrt{\frac{p_{exp}(1 - p_{exp})}{n}}$$

Results of this experiment yield a p-value = 0.0714. As $p > 0.05$, this result is not statistically significant, and the null hypothesis H_{05} is not able to be rejected. This indicates that the learning percentage is not decidedly as high as Class.ly creators expected, also indicating that there remains a high temporal load on users even as they perform the same actions repetitively within the application.

Graphical representation of learning percentage over two iterations of meetup creation can be seen below

Figure 5: Learning Percentage on Meetup Task



It is important to note here that the statistical results presented are drawn from a relatively small sample size (ten participants). Regardless, based on the differentiated background of users and for conciseness of testing, the Central Limit Theorem (CLT) is assumed to apply.

Before concluding the user testing, participants were asked to complete an exit survey from which the administrators were able to glean some interesting qualitative information.

- The most valuable functionality as perceived by participants is the easy meetup scheduler.
- The instant-message type interface was applauded and important to implement as opposed to the email-like interface within Blackboard and Piazza
- The participants want the meetup scheduler suggest proper times to meet based on external calendars.

V. Discussion

The user testing of Class.ly and subsequent revealed some interesting benefits, and many improvements to be made on the system before a production version would be released. First and foremost, based on the studies above, the creators of Class.ly are very pleased with the results indicating the comparative intuitiveness of their interface as opposed to competitors. Although Class.ly implements far fewer features than both Blackboard and Piazza, the way they are implemented- with Vanderbilt undergraduates in mind- allowed for highly optimized communication, as shown by the time on messaging task studies. Sending a message on Class.ly is approximately 85% faster than Piazza and over 100% faster than on Blackboard. This functionality and speed is absolutely essential for the target user demographic and psycho-graphic as communication is often on the go.

Additionally, with a standard deviation of time on the messaging task approximately 3x lower than that of Blackboard and Piazza, it is evident that there is an *extremely* well-defined critical path to complete specific activities on Class.ly. This is primarily due to the limited number of options inherent in the execution of the implementation; however, the application is designed with this simplicity in mind- deeming this low standard deviation to be a huge success. Users of the application would not be “wandering” around the side-drawer or other navigational features; however, they are directed to their desired interface with very little trouble and in a very few number of clicks.

The place in which expectations fell short as to the intuitiveness of Class.ly is the scenario where a user creates their first meetup. In this scenario, the average time on the task was much higher than expected, and there was a very large standard deviation. This standard deviation is primarily due to a specific near-outlier in the dataset; however, in such a small sampling, the point remains significant. In attempting to complete this task, users a required to navigate to a specific group within a specific course and use the navigational tabs to find the meetups. From there they must create a meetup with specified parameters. Most users did not have difficulty creating a mental model to lead them to a specific group; however, they could not find the meetup tab in order to navigate to that interface. The main reason for this is believed to be that the user testing took place on a desktop device. Tabs in a traditional browsing context are not commonplace and are rarely used. This likely contributed to users not looking to the bottom of the browser for the navigation mechanism (increasing their time on the task). On a mobile device, this result would most likely differ as tabs are the expected navigation mechanism. This difference has been noted by the Class.ly creators and would be addressed by changing the interface based on the browsing context. If the device being used was mobile, the tabs would persist; however, in a desktop interface browser, the interface would be rendered with the messaging and meetup tabs either in the side-drawer or as options in the top right corner of the screen (one of the first places a user looks for options).

The other very interesting result from the study was the higher than expected learning time to create a meetup in Class.ly. While the creators expected a mental model to take shape after just one iteration of creating a meetup, leading to a low memory load on users and a huge decrease in the time on task, this was not the case. Yes, there was a significant reduction in the time users spent; however, the increase was not statistically significant when the null hypothesis was that the learning percentage would be less than 50%. This means that either users need more training to develop their mental model, or the process of creating a meetup is not exactly clear. The Class.ly team believes the first case is more pertinent and an easier solution. There are three main ways to implement this solution

- Do nothing and allow users to build the mental model over time. Consequences here include high level of frustration or misunderstanding of how the system is meant to work.
- Release a video explanation of the functions and operation of the application on the informational website paired with the production application. Consequences here are the development and production time of an additional website and instructional video.
- Integrate a tutorial feature into Class.ly itself such that upon the first usage of the application, there is a series of 4-5 guidance screens. This may be disregarded by users; however, for some it would help build the mental model via interaction and prompting users to take certain “test actions”. Consequences here are of course development time (less than development of a completely new website), and anger as users would not be able to get started on the application right away.

These main points gleaned from the user testing demonstrate some of the successes and failures of the current implementation of Class.ly. Clearly the interface performed well when compared to counterpoints; however, this is likely due to the limited comparative functionality. Despite this success, Class.ly fell short in proving the application can be learned without much cognitive effort. Regardless, the current implementation is still a fantastic first step in what could

become a production application reducing the complexities in collaboration among Vanderbilt undergraduate students.

VI. Conclusion

The user testing of Class.ly revealed several key comparative performance benefits of the application against its counterparts, as well as several lacking features of the application itself. Users were generally impressed with the development effort and implementation; however, there was much to be desired. The desirable aspects were both functional (calendar integration) as well as stylistic (tabs in a desktop browser). Regardless of the current pain-points, the initial implementation of Class.ly is a sufficient building block for future versions.

The current implementation strives to account for modern design principles: mobile-first, intuitive navigation, low memory-load on users, and accessibility. While many of these are achieved in the current interface, additional design work is hugely necessary in terms of quickness to build a mental model and be notified of intra-applicaiton events (communication notifications). All of these features must be sufficiently thought through as they are critical to the success of any application to avoid annoyance, yet inform users of necessary information.

Specifying, designing, and building Class.ly has been a unique challenge this semester as the development lead-time and resources were several constrained. This led to few, semi-functional features as opposed to a fully flushed out application. Regardless, the functionality that has been completed are clearly most needed by the target demographic and serve their purpose with clarity.

References

[1] Kovach, S. (2011, March 10). 7 Reasons Why GroupMe Is The Best Group Messaging App Right Now. Retrieved November 21, 2015, from http://www.businessinsider.com/why-groupme-is-the-best-group-messaging-app-right-now-2011-3

[2] Create incredible apps. (n.d.). Retrieved November 20, 2015, from http://ionicframework.com/

[3] Get Started Fast. (n.d.). Retrieved November 20, 2015, from https://cordova.apache.org/

[4] Cocoa Core Competencies. (n.d.). Retrieved November 21, 2015, from https://developer.apple.com/library/ios/documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html

[5] NoSQL Databases Explained. (n.d.). Retrieved November 22, 2015, from https://www.mongodb.com/nosql-explained

Appendix A

This appendix includes the introduction script read by user testers to testing participants, the introduction survey, exit survey, and testing results collected during user testing. This a remote appendix and the necessary documents can be found in the “Final_Report” directory of the repository hosted at https://github.com/masiamj/CS3892-HCI-Class.ly.