Spring 2019

Take Note: An Investigation of the Impact of Human Computer Interaction on the Line Note Taking Process in Theatre

Rene Borr
Rollins College, rborr@rollins.edu

Take Note

AN INVESTIGATION OF THE IMPACT OF HUMAN COMPUTER INTERACTION ON THE LINE NOTE TAKING PROCESS IN THE THEATRE

RENÉ BORR

A THESIS PRESENTED TO THE FACULTY OF ROLLINS COLLEGE IN FULFILLMENT OF THE REQUIREMENTS FOR HONORS IN THE COMPUTER SCIENCE MAJOR.

Table of Contents

1. Abstract

Most technology used in the theatre was not originally designed for use in a theatrical setting. A majority of technology that we see on the stage has been adapted from another field - be it music, lighting or construction - to fit the production's specific needs. The very nature of theatre is that every performance is unique, which establishes one of the main challenges when creating technology to be used in the theatrical setting.

This paper discusses the design, creation and testing of a program to help stage managers take line notes during rehearsal for a theatrical production. We will begin with an outline of the problem as well as providing background information about theatre productions. We will then discuss related works, followed by a discussion of the design of the system, testing and evaluation of the system, and the final results of the study.

2. Problem and Background Information

Before a play or musical can be seen by an audience, it must go through a rehearsal process. Every day, the cast and directors of a production will meet and work on various portions of the show. Once the show has been completely rehearsed, the production will begin having full runs. A full run of a production occurs during a rehearsal when the show is performed in its entirety without any of the technical elements such as lighting, sound, or costumes.

The stage manager is in charge of ensuring that the production as a whole goes smoothly by working alongside the actors, designers, and technicians. During a performance, the stage manager oversees the calling of all cues that occur in a show. A stage manager will call a cue by talking into a headset connected to other crew members around the theatre, prompting the light, sound, or set to change. During rehearsal, the stage manager is in charge of notating all important information and distributing it to the necessary parties in a timely manner. If the rehearsal consisted of a full run, then the stage manager takes line notes and ensures that all actors receive them before the following rehearsal.

Line notes are notes taken by the stage manager and indicate what an actor said on stage versus what they should have said according to the script. This can be done in multiple ways, including manually highlighting portions of a digital script that correspond to the phrases said incorrectly, or notating them on paper, with a transcript of the line said incorrectly.

Unfortunately, all current methods used are not fast or efficient enough to make the process of taking line notes simple for the stage manager. It is nearly impossible to keep up with live speech while also cataloging mistakes. Additionally, after rehearsal, stage managers will spend hours organizing and compiling these notes for distribution, only to then repeat the process the following night.

The software system outlined in this thesis document attempts to augment the process of taking line notes for the stage manager by providing a fast and efficient system for not only taking line notes, but by automatically organizing them for distribution. This will allow the stage manager to take more accurate line notes and save them time post rehearsal.

Next, we will discuss other software in use in theatre environments (Section 3) and provide an overview of the structure and organization of our software (Section 4). Then we will present an exploratory pilot study evaluating the software (Section 5). Finally, we will conclude with a discussion of the results of the study and future improvements (Sections 6 & 7).

3. Literature Review

There is little computing technology developed for theatre in comparison to other fields. This literature analysis will discuss other studies conducted to develop software for live theatre performances as well as the organizational software developed specifically for theatrical productions. It will then analyze software developed for live captioning. Finally, we will discuss current software that is commercially available. By analyzing the following literature, we will be able to better understand the deficit in the industry that this system fills. While other programs

are designed to help aid designers and stage managers, the program outlined in this study is the only one that helps aid the process of taking line notes.

3.1. Theatre Design Software

One of the most challenging aspects of a theatre production is the collaboration between the different artists involved such as costume designers, set designers, directors and actors. Theatre is inherently a collaborative artform and requires that all parties communicate effectively, or the production as a whole will suffer. There have been multiple systems developed to assist with this specific problem which are discussed in this section.

A virtual reality system that was developed at Keio University was implemented to help theatrical designers collaborate over long distances (Horiuchi, Inoue & Okada, 2012). Additionally, it helped designers visualize aspects or a production that may not be possible otherwise, such as lighting design and abstract set designs, by utilizing a tabletop projection system. This interactive tabletop projection system utilized miniature figures to represent actors and other pieces of the stage to ensure that all designers in a production have a common understanding of design choices throughout the process. This system also helped create a cue sheet for use in the production. A drawback of the system was the inability to truly customize the software for the individual space. For example, lighting design concepts could not be truly expressed since the lights, colors, and lenses available on the stage were not always available for virtual representation, thus making this product not as viable for lighting designers. However, both the interactive tabletop display and the cue list were helpful for set designers and allowed them to express how the actors could move through space.

Another system, called the digital playbook, is an digital script user interface that allows both actors and designers to visualize a script in a more cohesive fashion (Sinclair & Ruecker, 2019). The lines and the characters were set up in graphical user interface, which allows the user to select and scroll through different scenes. The user can easily navigate between scenes and acts without having to flip through physical pages in the script. A unique facet of this software is its

ability to represent blocking. During certain scenes in a script, small dots appear on the screen with a name attached to them. These dots move around in correlation with the blocking notated in the script or by the user. This system allows the user, usually an actor or a director, to customize the movements and see them from the top down. By using this system, the actors and directors can experiment with different movements and placements before the rehearsal process actually starts.

3.2. Theatre Performance Software

Theatre performance software is similar to theatre design software in that it actively aids the theatrical production as an art from. However, theatre performance software is always viewed onstage, and it an active piece of art in the production. It aids the live performance in some way. Most theatre performance software systems are developed on a show-by-show basis and have no real use once that specific show has closed. Technology developed for theatre performances must be adaptable since it will be used in a live setting. Mistakes can happen and things will always change in live theatre. This section will discuss some theatre performance software systems developed in order to have a greater understanding of their effects on the actors and the production as a whole.

One such example of theatre performance software is a projection system for a production of *Dot and Kangaroo* developed at the University of Technology, Sydney (Bluff & Johnston, 2017). This interactive projection system allowed the actors to physically 'move' the projections behind them during the play. The main goal of this system was to supplement the need for verbal storytelling and allow a more visual story to be told. The survey done on the projection system showed that the scenes that utilized the projection system were considered more impactful. They concluded that scenes can more accurately portray the core message of the piece by using the projection system developed.

Another example of software being utilized for performance was a production staged at Michigan State University called *Transitions* (Owes, Dobbins & Rebenitsh, 2013). In this

production, live dancers were paired up with digital counterparts that were projected on screens around the stage. A unique aspect of this performance was the ability for the audience to interact with the projections on stage using their mobile devices. The audience members would "toss" digital dancers around using a swiping motion during certain portions of the production. Additionally, the audience members would control virtual birds on set by tilting their smart device back and forth. The main challenge faced by the designers in this production was encouraging audience participation. The developers had to ensure that users could use the application on their mobile devices without intervention or instruction from the people on stage.

3.3. Live Captioning

While not directly related to the theatre, the use of software to create live captions for those who either are attempting to communicate in a language that they are not fluent in or who have a hearing impairment can provide beneficial information when analyzing the most efficient ways for stage managers to do a similar thing when taking line notes. In this section we will discuss different systems developed and studies conducted to understand their success level as well as what we should include in our interface.

A research study done at the Toshiba Corporation attempted to distinguish the most efficient way to provide live captioning for interaction where a speaker has limited knowledge of the language used, that would maximize understanding (Shimogori, Tomoo, & Tsubio, 2010). The researchers discovered that if captions were shown two seconds before the person spoke, comprehension would improve. However, if the caption is shown approximately two seconds after the person on screen has spoken then comprehension would decrease. This finding was especially true if they are communicating in a language that both users speak, but one does not understand as fully. This is a problem, since it is impossible to provide captions for live conversation before words are actually said.

3.4. Commercial Products

When analyzing current commercial products that are on the market, it is important to recognize that these products were not the result of formal research, and therefore have no statistics about their usability or overall success. These products are widely used, probably more than the systems outlined above, and it is important to investigate their functionality.

Virtual Callboard, developed by Empty Space Technologies Inc, is an online based software used by both designers and actors ("VirtualCallboard", 2019). Virtual Callboard is designed to be an online database for all important documents in a production including designs, drafts and calendars. By allowing all departments in a theatrical production to stay organized and in-sync, Virtual Callboard provides a valuable service to its consumers. Another example of an all-around organizational software is called Propared. Propared, developed by Propared LLC, is a theatrical management software similar to Virtual Callboard ("Propared", 2019). Propared allows the user to create mega-calendars and lists and easily store valuable documents in one easy location. A major difference between Propared and Virtual Callboard is that Virtual Callboard is a web-based system, while Propared is locally stored.

Some software packages were designed to specifically help theatrical designers keep track of stock. Since plays and musicals often require specialized items, it is easy for designers to lose track of what exactly they have available at any given time. StageStock, also developed by Empty Space Technologies Inc, is designed to help with this specific problem ("StageStock*", 2019*). StageStock stores digital records of all physical items available. It then allows users to search through the digital archives to find a required item. This software also allows designers to compile a 'pull list' or a list of items to find and remove from inventory. Since they can view what an item looks like before they remove it from stock, they can find required items in less time.

The stage manager is responsible for calling cues for all movement on stage that is not the actors. One piece of software developed to help with this process is called Cueist, designed by

Josh Epstein ("Cueist", 2019). Cueist is an organizational software for stage managers that allows stage managers to make a prompt book compiled from all other departments involved in the theatrical production. This is helpful as previously the stage manager had to do this process by hand using pdf editing software. Another software designed to help with cues is a program called QLab, developed by Figure 53 (*"Figure 53|QLab"*, 2019). QLab is a program that compiles all cues for a production and works with the theatres lighting and sound system to play them when an operator presses go. This software helps the operator stay organized and follow the script with more ease in addition to having a more accurate prompt book for the stage manager.

Stage Write is a software system similar to Cueist and Virtual Callboard, as it is an organizational software designed for use in a large-scale theatrical production (“Stage Write”, 2019). A unique aspect of Stage Write is its ability to track blocking and movement on the stage. The user can add blocking notations and movements directly to an imported script and can track each actor simultaneously.

Products like the ones outlined in this section can prove to be invaluable when organizing large scale theatrical productions. However, none of the systems found had functionality to assist with the process of taking lines notes. While our software system can stand alone, its functionality could be especially helpful if used in tandem with other wholistic organizational software like the ones outlined above.

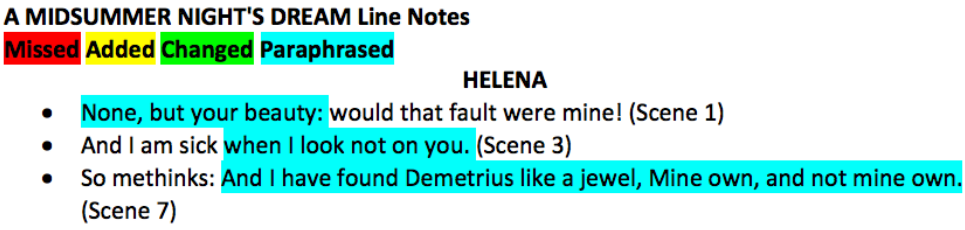
4. Design of System

This section will discuss the construction of the software system for taking line notes. It will outline the process in which the requirements for the system were created as well as explain the front-end and back-end design.

4.1. Formative Evaluation

Before designing the overall system, a series of preliminary interviews were conducted. A total of six stage managers were interviewed and asked questions about their experience taking line notes. Each of these stage mangers had stage managed at least one full production at Rollins College.

During the interviews, it was concluded that a majority of stage managers followed a similar method for taking notes. The stage manager would have a digital version of the script and highlight specific lines or words. The color of the highlight would correspond to the type of error made. Figure A shows an example of line notes taken for a production of *A Midsummer Night's Dream*. The notes contain words and phrases highlighted in blue to show that they were paraphrased. The stage manager might add additional notes below the line to clarify how the line was said incorrectly. Finally, the stage manager would either remove all lines with no errors or re-organize the missed lines by the character who spoke them before distributing the notes for the actors to review.


A MIDSUMMER NIGHT'S DREAM Line Notes
Missed **Added** **Changed** **Paraphrased**

HELENA

- None, but your beauty: would that fault were mine! (Scene 1)
- And I am sick when I look not on you. (Scene 3)
- So methinks: And I have found Demetrius like a jewel, Mine own, and not mine own. (Scene 7)

Figure A: An example of line notes taken for a production of A Midsummer Night's Dream

Once the interviews were completed a list of system requirements were compiled. These requirements were broken down into two categories based on their priority: essential functionality or clarity of notes.

The requirements for essential functionality are as follows. The system must be able to:

- parse and format a text-based script for display.

- display a script in a traversable manner which allows the user to keep up with the play in real time.
- allow the user to quickly mark if a line was said incorrectly.
- export all notes in a readable format which can be understood by an actor performing in the production.

These requirements are the highest priority and include the minimum features in order for the program to serve its desired purpose. The program needs to be able to take in a script file, display it for the user, allow the user to add markings for incorrect lines, and export it for the actor to view once rehearsal is over. A majority of the stage managers that we interviewed used a PDF file format to distribute line notes to the actors. Unfortunately, a PDF is not easily created and edited using the Java programming language. It was concluded that the best compromise between programmability and usability would be to format the export file as a Microsoft Word document. This would allow the user to open the file in an editor and make additional adjustments as needed. While the system strives to provide notes in a format that would not require additional editing, providing the user with an editable document would enhance the usability of our program.

The list of user requirements for clarity of notes is as follows. The system should be able to:

- allow the user to select specific portions (words) within a line that were said incorrectly, not just the complete line.
- allow the user to mark different ways in which a line could be said incorrectly, preferably about five options with color coordination.
- allow user to write additional text notes that would be included in the export.
- allow the user to categorize how lines are exported, either by character or scene.

While the program would run and serve its main purpose without the above requirements, they are necessary to match current stage management practices. With the above requirements gained from our formative evaluation, we began construction of the software system.

4.2. Back-End Design

Following object-oriented principles, each of the elements in the script is represented by one class in the back-end code. This system was programmed in the Java language because it is object-oriented and allows for simple creation of graphical user interfaces.

The *Script* object is the main object for the source code and allows for interaction between the user interface and the back-end of the system The *Script* keeps a list of *Scenes* and *Parts* that exist within the text and provides methods for accessing them.

The *Scene* object represents one portion of a complete *Script* and is responsible for maintaining a list of *Lines*. Additionally, a *Scene* knows its individual scene number and what act it is in if applicable. It was decided that acts would not be represented by its own object. Since most scripts only have two acts, and many only have one, it overcomplicates the system to incorporate the idea of acts as an object. However, the information would still be helpful for the user; therefore, the *Scene* holds a string representation of what act it is located in. The act number is stored as a string, not an integer, because some act numbers are represented by Roman numerals in some scripts.

A *Line* represents one line spoken by a specific actor onstage and maintains a list of *Words* which comprise the line. Additionally, a *Line* keeps track of the individual page number where it is located in the script. Correspondingly, a *Word* is a simple object, containing one string (the word that it's storing) and its *Status*, representing if the word was said correctly. *Line* has the ability to iterate through itself and mark a portion of its *Words* with a specific *Status* as defined by the user. *Line* also contains a pointer to the *Part* that spoke it in the *Script*.

The *Part* object exists outside of the individual *Scenes*. A *Part* keeps track of a character name, and what lines were said by the corresponding character. A *Part* is useful to track how many

lines an individual character or actor said incorrectly. A UML diagram of the source code is provided showing the relationship between the *Script, Scene, Line* and *Word* objects (Figure B).

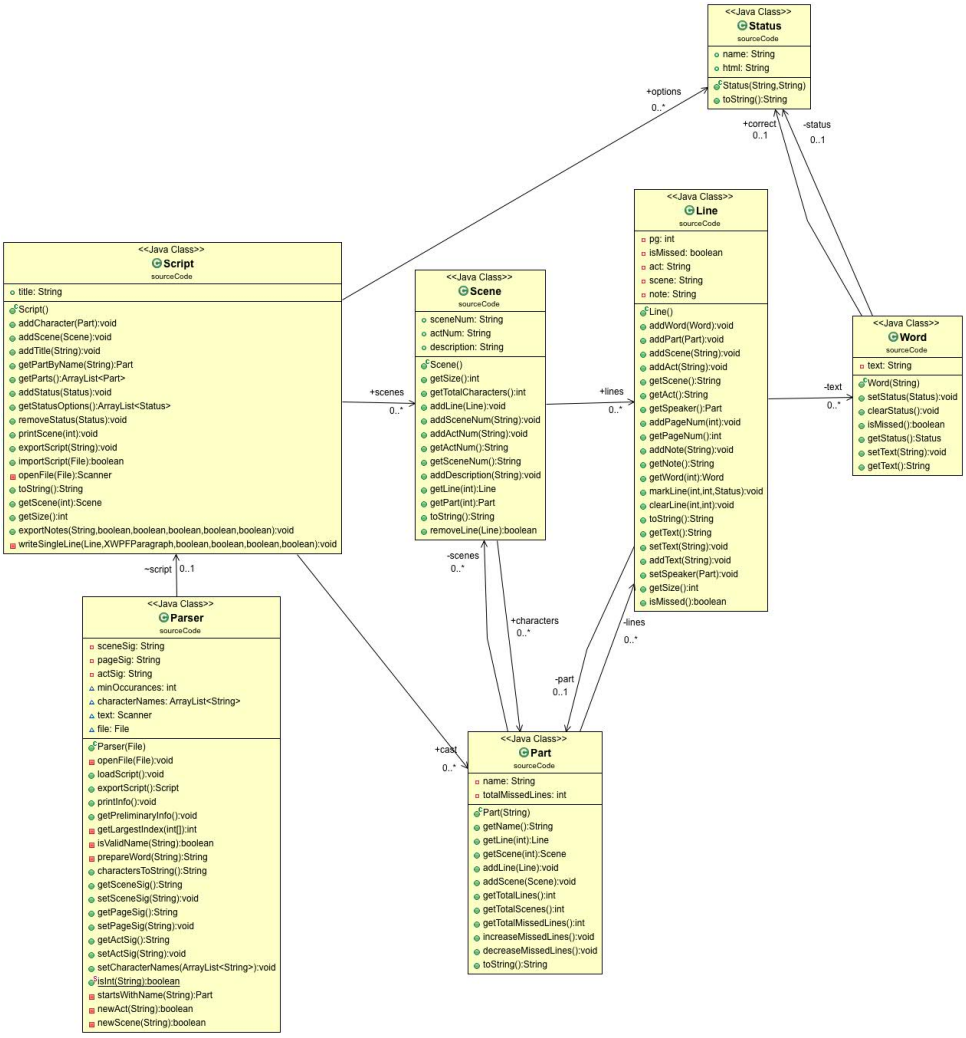


Figure B: A UML Diagram of back-end design

The *Parser* is the only object that exists outside of the *Script* Hierarchy. The *Parser* is responsible for parsing a text-based script file into a *Script* object. In order to parse the many different types of script formats that this program might encounter, the *Parser* is designed to pass through the script twice. During the first pass, the *Parser* attempts to determine what the names of the characters are and what delineators are used to define a change of page, scene and act within the script, after its first pass is verified by the user. the *Parser* separates the script into individual *Words*, *Lines*, *Scenes* and *Parts*, placing them into the *Script* on the second

pass. Once the *Script* has been built, the *Parser* has completed its job, and is not used again in the process.

4.3. Front End Design

The front-end of the system is separated into two distinct interfaces. The Notetaker Interface (Figure C) where the user will notate errors said by the actor, and the Parser Interface where the user can build and edit a new script file to be used in the system.

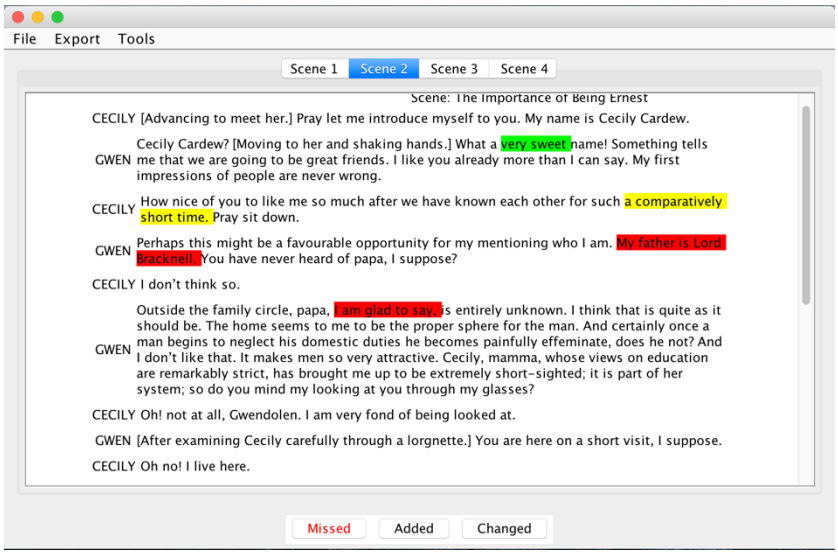


Figure D: Example of the NoteTaker Interface

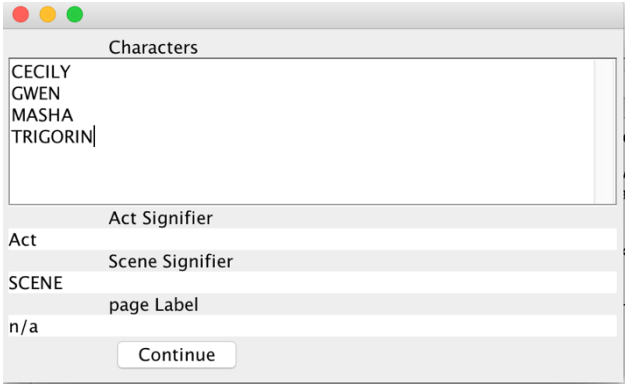


Figure C: Example of confirmation screen

Both the Notetaker Interface and the Parser Interface utilize the Script Frame component. Script Frame displays the script to the user and monitors all changes made. The Script Frame has two distinct "styles": editor and note-taker. These styles allow the Script Frame to be reused in different interfaces. The editor style allows the user to add, remove, and edit lines

and is used in the Parser Interface. The note-taker style allows the user to highlight words the actor said incorrectly and is used in the Notetaker Interface. The Color Panel, called the Status Frame in the code, is located at the bottom of the Notetaker Interface. It allows the user to switch back and forth between different colors or categories by clicking on the associated button.

The Parser Interface is split into two separate components: The Confirmation Interface (see **Error! Reference source not found.**) and the Script-Editor Interface. The Confirmation Interface allows the user to confirm information the *Parser* found on its first pass before it begins its second pass as discussed previously. By including this functionality, the user can confirm that the parser identified all of the information correctly, allowing the parser to be more accurate in building *Scenes*, *Lines*, and *Parts* on the second pass. The Script-Editor Interface looks similar to the Notetaker Interface and allows the user to edit different portions of the script by adding, editing and removing lines that may have been missed by the parser on the second pass. A UML diagram of the front-end design shows these relationships (see Figure E).

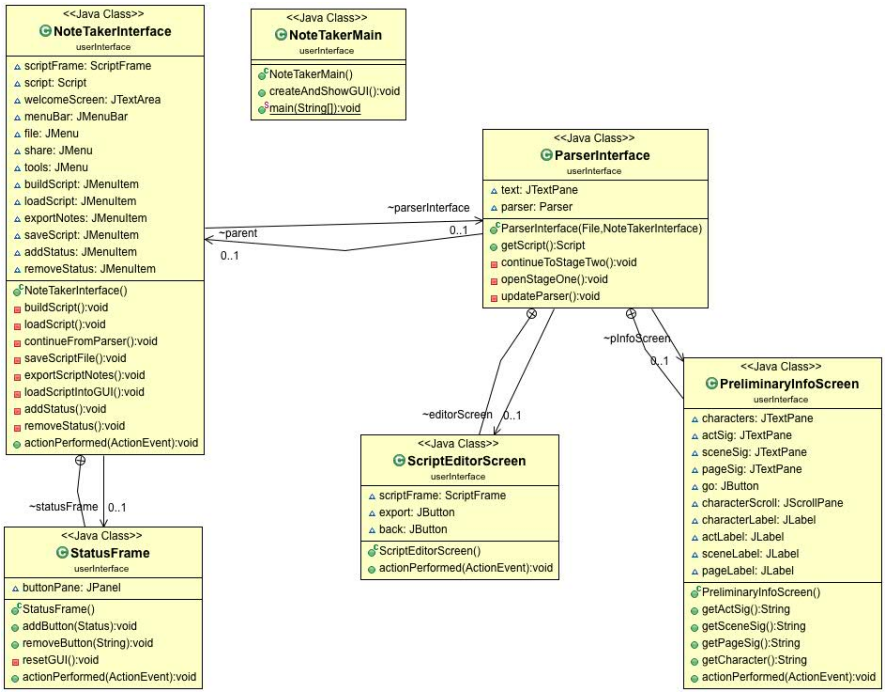


Figure E: UML diagram of front-end design

4.4. Script Import and Export

This system can save a script file after is has been parsed. Since the editing process can often be time consuming for the user, the system was designed with the idea that this should only need to be done once per script used by the stage manager. Once the user has built a script, the user has the option to export this script into a line note file (file extension ".ln"). On subsequent uses, the user can simply load the pre-built script into the interface, instead of going through the process of parsing and editing it once more. The program checks for files with the extension ".ln," and will not read them without this extension. The ability to read these files is located inside the *Script* object.

5. Evaluation of System

To understand the effectiveness of the software, we designed a study to assess the systems usability in a simulated theatrical environment. The nature of live theatre is that it changes every time. This would make it hard to compare results of the usage of the software from different productions. By simulating a theatrical environment, we can more easily compare and contrast data obtained from different users. Additionally, shorter testing scenarios would allow for more test users to participate.

For the study, we attempted to simulate live theatre as closely as possible, accounting for the many different types of performances in which a potential user might use this system. We narrowed the testing scenarios down to four categories. The test users would each attempt to use the software to take line notes during the four different types of scenes. Two of these scenes would be from comedic material while the other two would be from dramatic material. Comedic material tends to be performed at a faster pace than its counterpart. The contrast between comedic material and dramatic - which usually contains more pauses - is important when using this system. For both categories, we would choose one solo performance also known as a monologue, and one scene containing two actors. This would allow us to see how the user could adapt to one person speaking, versus alternating between two different characters. Each of these scenes would be approximately one to two minutes in length. By

using these four testing scenarios, we would be able to cover the widest range of possibilities. While we understand it is impossible to cover all potential scenarios, we chose ones that we felt would be most commonly seen by a stage manager.

A list of the scenes chosen, as well as the categories that they fall under are:

- Mabel from *An Ideal Husband* by Oscar Wilde (Comedic Monologue)
- Cecily and Gwendolen from *The Importance of Being Ernest* by Oscar Wilde (Comedic Scene)
- Madame Ranevsky from *The Cherry Orchard* by Anton Chekov (Dramatic Monologue)
- Masha and Trigorin from *The Seagull* by Anton Chekov (Dramatic Scene)

The scenes were then recorded using different volunteer actors from the Rollins College Theatre Department. We chose to use different actors for each scene, instead of repeating actors, to ensure that there was enough variation between the different scenarios and performances. The actors preformed each scene to the best of their ability from the script provided to them. Each scene was recorded with approximately ten errors of varying types. These error types fell into three different categories that could be easily simulated; missed, added, and changed. A missed portion of a line is highlighted in red and represents words that were not said by the actor at all. An added error is highlighted in yellow and represents words not in the script but was said by the actor. Finally changed errors were highlighted in green and represent a word or phrase that was substituted by the actor. For example, an actor might change “horrible” to “terrible” when preforming. In a quiet room, each of the four scenes was played in succession on a projector. The participants used our system and attempted to notate the errors that they perceived. We decided to only use the pre-determined set of three categories/colors instead of allowing each individual user to create their own. While this would not allow each user to explore the full extent of our program, it would allow us to more easily compare each user’s results with others.

This method of testing does have some drawbacks that should be addressed. First, each of the scenes that were used for testing was only one to two minutes in length. It did not accurately portray how the users would be affected by fatigue after taking notes for multiple hours as they would in a normal rehearsal setting. Additionally, because of the pre-scripted nature of the testing, certain error types could not be tested for, such as when an actor forgets a line and needs to be prompted by the stage manager. However, despite these drawbacks, evaluating the software as outlined above is an efficient way to understand the strengths and limitations of the systems design as compared to the more traditional way of taking notes.

Once the testing was completed, the data was analyzed to determine how well the participants preformed. The line notes from for each participant were compared to a correct version to obtain the total number of errors each participant was able to locate. If a participant located an error but marked additional words before or after it, or if the error was found but categorized incorrectly, we categorized the line note as correct. This is because in a real theatrical setting, the actor will still be able to use this information to understand that a line was said incorrectly. A side-by side-comparison of the correct notes and a participant's notes is shown in Figure F.

Additionally, each participant was interviewed with the following questions after using the software:

- What were your overall impressions of this system?
- How is this system similar or different to how you have taken line notes in the past?
- What did you like about this system, and what would you change?

With the data collected and the answers to the above questions, we were able to obtain solid impression of the overall success of the system.

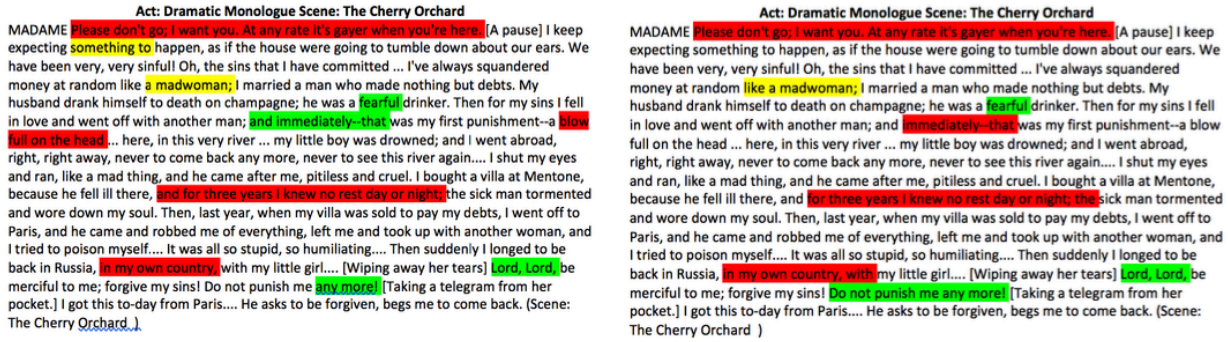


Figure F: The correct line notes for the scene (left) and the line notes taken by participant 2 (right)

6. Analysis and Discussion

Overall, the results from the study were positive. On average, the participants were able to find approximately 70% of all errors that existed in the testing scenarios. Since the testing pool for this study was small, we cannot assume that these findings will hold for a larger testing scenario. However, the statistics that were gathered from this study will be used to discuss the systems strengths and areas for improvement. Along with the statistics gathered, the qualitative feedback from the users will also be discussed to gain a holistic view of this system's strengths and weaknesses as well as to understand the users' impressions of the system.

6.1. Analysis of Statistics

As mentioned above, the participants as a group were able to locate 70% or the errors in all four videos. In the four testing scenarios there were a total of 41 errors, with 10 in each of the first three scenes and 11 in the last scene. On average, the participants were able to find 4.8 errors in the first scene, 7.4 in the second, 7.4 in the third, and 9.2 in the final scene. There was distinct improvement in performance by the user from the first scene to the final. In the first scene participants found 48% of errors, and in the final scene participants located 84% of the errors. This increase in correctness can be attributed to two factors: participants becoming

more familiar with the system and the speed at which the scenes were performed. The first scene was comedic while the second was dramatic. Since comedic pieces are often preformed faster than dramatic ones, it makes them harder to takes notes for. For a full breakdown of data gained from this study, reference Appendix A.

On average, participants had an easier time locating errors in the missed category and a harder time locating them in the changed and added categories. Overall participants identified 79% of missed errors, and only 60% of added and 63% of changed. This difference is probably due to the fact that the latter categories are not as concrete as the former. It is easier to see when an actor misses words in the script, as opposed to when they add or re-arrange words.

We must also take into consideration that stage managers are likely to use additional or different categories for marking errors than the ones given, and this functionality is provided by the system. All participating stage managers were required to use the same categories for the study for the sake of continuity. The categories of “Missed”, “Added” and “Changed” were decided upon because they encompass the breath of the different types of errors that a stage manager would see in most performances.

Overall, the statistics gained from the user study were promising as to the overall usability of the program. By the final scene, after the participants had received a brief walkthrough and six minutes of practice from the scenes prior, the participants averaged 84% correctness. While we cannot be certain if these findings would hold in a real-world study, this outcome is very promising. A previous familiarity with the script may change these outcomes. In a real production, the stage manager would be intimately familiar with the script beforehand. Table 1 shows the amount of errors each participant was able to find broken down by scene.

	SCENE 1 (10)	SCENE 2 (10)	SCENE 3 (10)	SCENE 4 (11)
PARTICIPANT 1	5	8	8	10
PARTICIPANT 2	4	8	7	10
PARTICIPANT 3	5	7	8	8
PARTICIPANT 4	6	8	7	9
PARTICIPANT 5	4	6	7	9
AVERAGE FOUND	48%	74%	74%	84%

Table 1: Breakdown of errors found by each participant

6.2. Usability and Strengths

As a whole, the feedback gained from the users was positive. Multiple users remarked on the fact that overall, the program was user friendly and intuitive to use. Participant 1 stated that the program "covers all the bases and it was pretty easy to use." Additionally, participant 3 said "I think this [the system] is really great. It can really help people."

The participants brought to light 4 specific strengths of the system from their perspective when compared to other methods of taking line notes:

- The User can click to move from scene to scene instead of scrolling.
- The User can click on the Color Panel to change categories.
- The User can double click to mark specific words.
- The User can export notes into a Word document.

With the ability to click to navigate from scene to scene instead of scrolling, the user can save time and more accurately track their location in the script. Participant 2 talked about the ability to change scenes by saying, "I like how you can flip back and forth between the scenes I think it is a lot easier to be able to see all the scenes instead of having to, like, scroll. … When taking line notes we have to scroll down to find things that takes a little bit long."

With the ability to use the Color Panel to change categories, participants were able to reduce the number of button clicks per note taken, thus speeding up the process as a whole. With a

Word document, the user would need to highlight the desired section, find the small button for highlighting, then navigate through the different colors to find the desired one. Additionally, due to the fact that the buttons on the Color Panel were marked with the category name and color, this system provided a reduced cognitive load in comparison to other methods. Specifically, users would not have to memorize what color was associated with a certain category as they would using general purpose software such as Word. Participant 3 said, “If we have a script, we will just highlight it in different colors. You have to highlight it, then go up and change the color. It takes longer than this program did because this has just the options and it will automatically make it the color you want if you click on the right thing [Color Panel].” Participant 4 noted that, “the hard part about highlighting in general is that … by the time I’ve switched the color over … I could’ve missed something, whereas in this [program], I just click on the word, and it already has the color… So, it cuts back on a lot of time-consuming steps in that regard.”

Many stage managers particularly liked the ability to double click individual words to mark them. This made it easier to mark minute details in a script and keep moving with the recording. Participant 2 remarked that, “I think [Highlighting] is especially easier when it’s just one word. When you can just double click on it is a lot easier.”

While not explicitly tested in the study, the participants appreciated the software’s ability to export line notes into a usable Word document. This is a functionality unique to this method of line note taking, and it saves the stage manager valuable time post rehearsal. Participant 3 said “[Compiling notes] is what takes the longest for me, so it’s really helpful that I don’t have to do that manually.”

6.3. Critiques and Improvements

In addition to the strengths of the system, most of the participants also mentioned critiques and improvements to enhance its usability.

Participants brought to light the following critiques for the program:

- The dropdown menu covers portions of the script at inappropriate times.
- The system lacks a backup in case of computer malfunction.
- Lighter colors are harder to see on the Color Panel.

First, multiple users remarked that the drop-down menu that appears when the user right clicks to highlight a portion of words blocks their vision of some of the remaining sentences. The users suggested that the drop down be above the curser, instead of below, allowing them to read ahead when taking notes. Participant 1 asked if, "the menu for marking could come above the lines, because it goes over the lines that are being said. As you're highlighting, the actors are still speaking, so you have to be reading while you're highlighting. If the menu could flip up, then it only covers lines that have already been spoken, and it's not going to interfere with any lines that are being said currently." Participant 2 mentioned that marking entire sentences was harder because of the dropdown menu. They said, "With a sentence it's not difficult. It's just when you're highlighting and right clicking, you're also trying to read at the same time. When you right click and the little menu comes down, it blocks a little bit of the words."

Some participants brought to light a concern that the software does not have a backup if the user's computer were to crash. Currently some stage managers use online programs such as Google Docs so that multiple stage managers can take notes simultaneously, and they would not lose progress if a personal laptop were to run out of battery or freeze mid show.

Finally, Participant 2 complained that the shade of yellow chosen was hard to see on the button panel (see **Error! Reference source not found.**). They said, "I like that the words light up in the [Color Panel] but 'added' is hard to read because it's a bright yellow. I can't see it with my eyes." To alleviate this problem the Color Panel could be changed to have the color show up as the border or background of the individual button, or simply use a more "golden yellow" as suggested by the participant.

7. Conclusion

In this paper we have discussed the design, implementation, and study of a software system to aid stage managers in taking more efficient line notes during the theatrical productions. We have designed a system that we believe not only maintains the level of accuracy from other methods but increases it. Additionally, our system saves the user valuable time by compiling notes automatically for distribution. After completing a user study, we found that on average, the users were able to find errors with 70% accuracy using our system. While some participants had feedback regarding ways to improve the system, their overall impression was positive. The software system outlined in this study has the potential to truly make the process of line note taking faster and more efficient for the user.

References

Andrew Bluff and Andrew Johnston. (2017). “Storytelling with Interactive Physical Theatre: A Case Study of Dot and the Kangaroo.” *Proceedings of the 4th International Conference on Movement Computing*, ACM, (pp. 19:1–19:8). *ACM Digital Library*, doi:10.1145/3077981.3078036.

QLab. (10 Oct 2018). "Figure 53 | QLab". Retrieved from https://figure53.com/qlab/

Cuelist. (10 Oct 2018). "reinventing collaboration for theatre and live events". Retrieved from https://www.thecuelist.com/.

Yosuke Horiuchi, Tomoo Inoue and Ken-ichi Okada. (2012). “Virtual Stage Linked with a Physical Miniature Stage to Support Multiple Users in Planning Theatrical Productions.” *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces*, ACM, (pp. 109–118). *ACM Digital Library*, doi:10.1145/2166966.2166989.

Charles B Owen, Allison Dobbins and Lisa Rebenitsch. (2013). “Theatre Engine: Integrating Mobile Devices with Live Theatre.” *Proceedings of International Conference on Advances in Mobile Computing & Multimedia*, ACM, (pp. 378:378–378:386). *ACM Digital Library*, doi:10.1145/2536853.2536867.

Propared. (10 Oct 2018) "Production Planning Software Revolutionizing how Organizations Manage Show Logistics and Streamline Communications." Retrieved from https://www.propared.com/. Accessed 10 Oct. 2018.

Nobuhiro Shimogori, Ikedo Tomoo and Sougo Tsuboi. (2010). “Automatically Generated Captions: Will They Help Non-Native Speakers Communicate in English?” *Proceedings of the 3rd International Conference on Intercultural Collaboration*, ACM, (pp. 79–86). *ACM Digital Library*, doi:10.1145/1841853.1841865.

Stefan Sinclair, Stan Ruecker, Sandra Gabriele, and Anthony Sapp. (2006). "Digital Scripts on a Virtual Stage: The Design of New Online Tools for Drama Students." *Proceedings of the Fifth IASTED International Conference on Web-Based Education* (WEB 2006) Puerto Vallarta, Mexico: ACTA Press.

StageStock. (10 Oct 2018). "StageStock - Cloud-Based theatre inventory for Props, Costumes, and Lighting and More." Retrieved from http://www.stagestock.com/.

Stage Write. (4 May 2019). "Capture Creativity with Stage Write." Retrieved from https://www.stagewritesoftware.com/.

VirtualCallboard. (10 Oct 2018). "VirutalCallboard - Online Stage Management and Production Management." Retrieved from http://www.virtualcallboard.com/.

Appendix A

Below is a table showing the breakdown of how each participant preformed in the test study.

	Missed	Added	Changed	All
Scene 1				
Total	4	3	3	10
Participant 1	2	2	1	5
Participant 2	3	1	0	4
Participant 3	1	3	1	5
Participant 4	3	1	2	6
Participant 5	2	2	0	4
Average	2.2	1.8	0.8	4.8
% Found	55%	60%	27%	48%
Scene 2				
Total	4	2	4	10
Participant 1	3	1	4	8
Participant 2	3	2	3	8
Participant 3	4	1	2	7
Participant 4	3	1	4	8
Participant 5	3	2	1	6
Average	3.2	1.4	2.8	7.4
% Found	80%	70%	70%	74%
Scene 3				
Total	4	2	4	10
Participant 1	3	1	4	8
Participant 2	4	0	3	7
Participant 3	4	1	3	8
Participant 4	3	1	3	7
Participant 5	3	1	3	7
Average	3.4	0.8	3.2	7.4
% Found	85%	40%	80%	74%
Scene 4				
Total	5	2	4	11
Participant 1	5	1	4	10
Participant 2	5	2	3	10
Participant 3	5	1	2	8
Participant 4	4	1	4	9
Participant 5	5	2	2	9
Average	4.8	1.4	3	9.2
% Found	96%	70%	75%	84%
All Scenes				
Average	79%	60%	63%	70%