Manual for the CASAA Application for Coding Treatment Interactions (CACTI): A free, open-source program for rating client and provider speech.

This manual is currently under revision (15 May 2012). Do not reference or publish without permission.

Kevin A. Hallgren, Lisa H. Glynn, Jon M. Houck, and Theresa B. Moyers

Center on Alcoholism, Substance Abuse, and Addictions

University of New Mexico

casaa.unm.edu

Overview

The CASAA Application for Coding Treatment Interactions (CACTI) was developed as an instrument to facilitate accurate, sequential parsing and coding of human verbal interactions. CACTI has a simple interface for users to partition verbal interactions into discrete utterances that may then be coded by individual raters. Coding data for each utterance, including the beginning and ending times, byte numbers, and rater codes, are stored sequentially in a tab-delimited text file that may then be processed for data analyses in statistical packages like GSEQ, SPSS, R, and SAS.

CACTI provides clear advantages over other coding software. First, CACTI allows a user to code complex verbal behavior without the need for the speech to be transcribed manually, which can save money and time. Second, CACTI saves all data digitally, which can be manipulated directly using other spreadsheet or statistical packages, thus reducing the cost and error associated with the manual data entry necessary for transcripts coded on paper. Third, CACTI retains the temporal sequence of codes, thus allowing researchers to assess not only the frequency of behavior counts, but also the patterns of sequences in which the behavior occurs. Fourth, CACTI was designed for use in studies where audio recordings are initially parsed into discrete units, which are subsequently coded by trained raters, a feature which other software packages do not offer.

CACTI is free and open-source, per the terms of the GNU Public License: http://www.gnu.org/licenses/gpl-3.0.txt. Anyone is free to download, modify, and redistribute the software and source code, which are available at https://code.google.com/p/ctcsu-player/. Note that this software is unsupported by UNM CASAA.

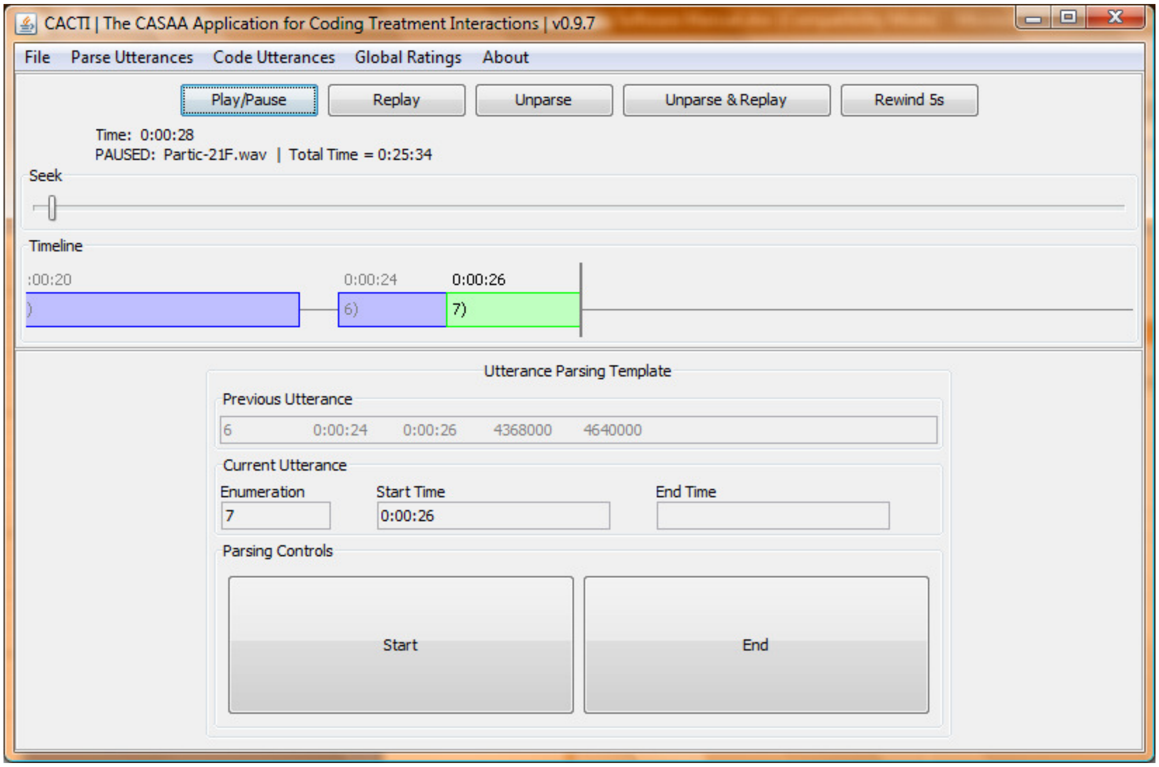
CACTI was developed by Alex Manuel, a computer scientist at the University of New Mexico, and Carl Staaf, a consultant at Robot Super Brain, in conjunction with Drs. Theresa B. Moyers and Jon Houck of the UNM Center on Alcoholism, Substance Abuse, and Addictions. Funding for the development of CACTI was provided by NIDA R01DA021227.

Top 10 reasons to use the CACTI

1. Reduces money and time spent on transcription of lengthy interactions
2. Stores all data digitally, thus reducing error and time associated with manual data-entry
3. Is open source software, and thus free to use, edit, or redistribute for anyone and no entity profits off of its use
4. Store sequential information for each utterance, allowing for more sophisticated data analyses than simple frequency counts
5. Is easy to use
6. Saves paper necessary for printing lengthy transcribed conversations
7. Is adaptable to be edited to work with most coding systems, including those that use behavior counts and global Likert-type ratings
8. Is developed by social scientists, for social scientists
9. Is Java-based, and thus should run nearly uniformly across operating systems
10. Coded interactions are pre-parsed, and thus the same pre-parsed units of speech can be coded by multiple coders to assess utterance-by-utterance inter-rater reliability that may be necessary for sequential coding studies.

Interface

Parsing GUI



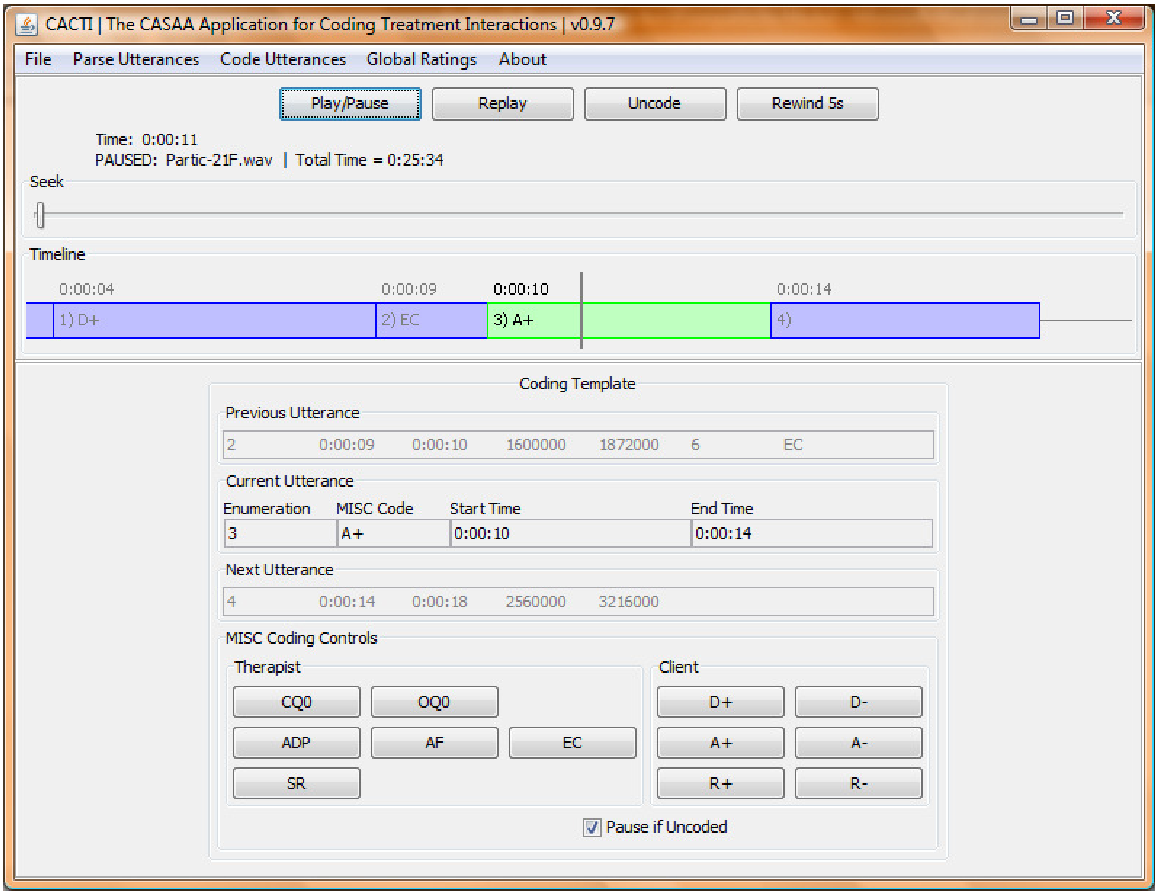
CACTI | The CASAA Application for Coding Treatment Interactions | v0.9.7

Time: 0:00:28
PAUSED: Partic-21F.wav | Total Time = 0:25:34

Previous Utterance: 6 0:00:24 0:00:26 4368000 4640000

Current Utterance
Enumeration: 7
Start Time: 0:00:26
End Time:

Coding GUI using MISC 2.5 codes.



Time: 0:00:11
PAUSED: Partic-21F.wav | Total Time = 0:25:34

Previous Utterance

2 0:00:09 0:00:10 1600000 1872000 6 EC

Current Utterance

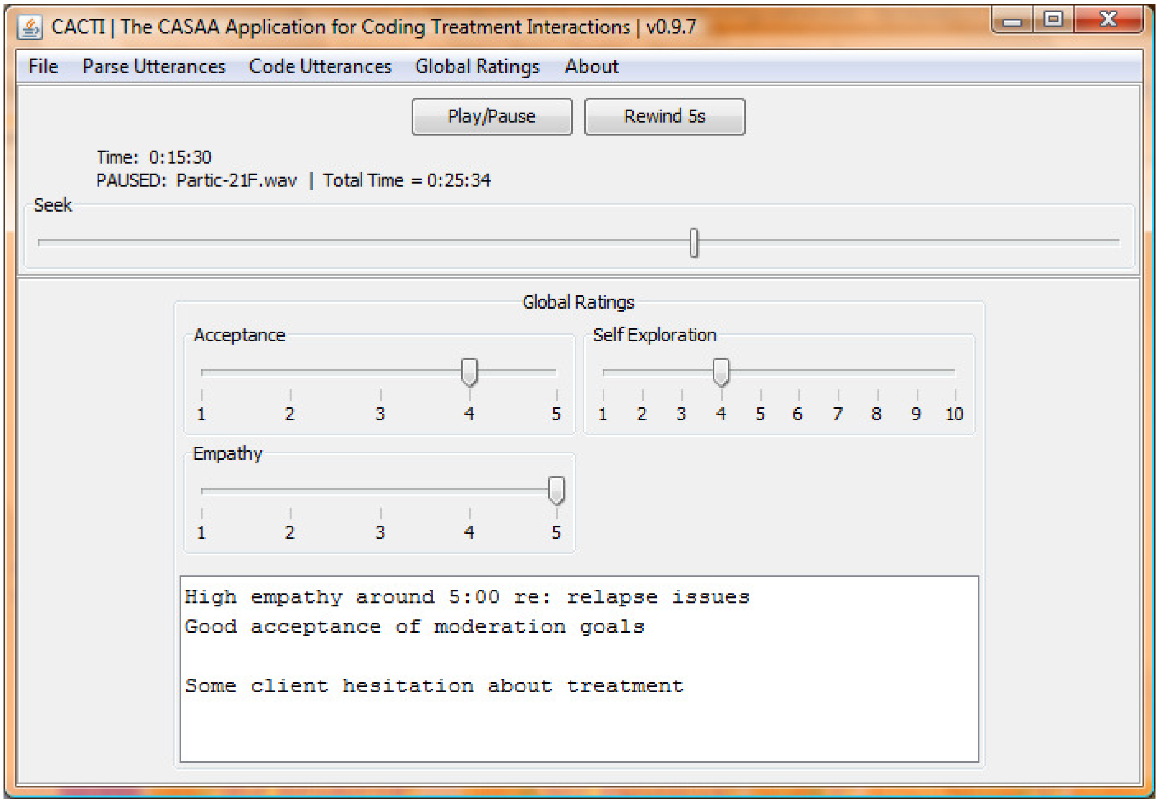
Enumeration	MISC Code	Start Time	End Time
3	A+	0:00:10	0:00:14

Next Utterance

4 0:00:14 0:00:18 2560000 3216000

☑ Pause if Uncoded

Global Ratings GUI



```
High empathy around 5:00 re: relapse issues
Good acceptance of moderation goals

Some client hesitation about treatment
```

Basics

Installing the Software

Proper use of the software requires that machines are updated with the most current version of Java, which can be downloaded for free at http://www.java.com. The latest software release may be downloaded at http://code.google.com/p/ctcsu-player/downloads/list and the source code is available at http://code.google.com/p/ctcsu-player/source/checkout.

To use CACTI, download the file CACTI.zip and extract the file on your local disk in a location of your choice. The zip file contains the following files:

- An executable file that opens CACTI for parsing, sequential coding, or global coding.
- An XML file that allows users to adapt CACTI to other coding systems (explained below).
- An images folder that contains a JPG graphic that appears as the splash screen upon opening CACTI.
- The operation manual and user configuration guide, included in the CACTI package, which are optional help texts that provide an orientation to configuring and using CACTI.

To open the software, double click on the CACTI.jar file.

General Format of the Software

Three basic coding procedures are available: parsing, behavior coding, and global coding. An audio file must first be parsed before it is behavior coded; however, global scoring may be done irrespective of whether a file was parsed or coded, and the sequence in which global scoring is performed can thus be determined by the research protocol for the given study.

`Parsing:` Parsing, or unitizing, refers to the process of breaking verbal information into basic, discrete units (called utterances or parses). For example, a user could parse the dialogue "How much did you drink?" "About 2 glasses of wine" into two discrete utterances reflecting the speech unit of the interviewer and the speech unit of the interviewee. A user may also wish to parse a single stream verbal behavior into multiple utterances if it reflects qualitatively different types of language. For example, "You must miss having your children at home. When was the last time you saw them?" could be parsed into two utterances that represent the two different expressions in the single stream of speech.

`Behavior coding:` Before a file is coded it must first be parsed. During the behavior coding process, a single categorical language code is assigned to each utterance. Each utterance is assigned a code.

The codes used during the behavior coding process will depend on the coding system employed for a given research study. For example, using a coding system may dictate that the utterances "[1] You must miss having your children at home. [2] When was the last time you saw them?" would be coded as a [1] therapist complex reflection, neutral and a [2] therapist closed question.

`Global ratings:` Global ratings are meant to capture the gestalt of an entire session or recording, using multiple ratings provided on Likert-type scales. They may be adjusted throughout the listening period of a recording or may be assigned at the end.

`Playing an audio file without coding:` The CACTI software may be used to play an audio file without coding or parsing (akin to a simple media player). To play an audio file without the intention of coding, select File > Load audio file… and select the audio file you wish to play.

`File formats for behavior coding:` Waveform Audio Format (WAV or WAVE) files have been tested using and are found to work for all aspects of the software. It is recommended that WAV files are used for all aspects of coding. WAV format is advantageous because it contains a constant number of bytes per second and works well with the software, which tracks both the time stamp and the byte number for parsing and behavior coding. Other formats, such as MP3, do not have a constant number of bytes per second, and parsing and behavior code decisions may be imprecise and/or inconsistent when using MP3s.

Customizing CACTI for your coding system

Before behavior coding, a user will need to customize CACTI to read the coding system they're using for their study. CACTI allows users to customize behavior codes and global ratings.

The GUI can be changed by editing the userConfiguration.xml file included with the software. This file should be edited in Notepad or another simple text-editing program, NOT with a word processor such as Word or WordPad. The userConfiguration.xml file is loaded automatically when CACTI starts, and therefore CACTI must be restarted for any changes in userConfiguration.xml to take effect. Before making changes to this file, we recommend saving an unchanged copy of this file that the user may revert to in case of any unidentifiable errors.

A list of keywords that allow users to modify CACTI via userConfiguration.xml is included below.

XML keyword	Description	Arguments
`<code>`	Declares a behavior code	`name`: Character identifier appearing on interface and .casaa file `value`: Numeric identifier appearing in .casaa file
`<button>`	Places a declared behavior code on the interface as a button	`code`: name of code to appear (must match a name in `<code>`) `key`: shortcut key, e.g., "s", "ctrl s", "ctrl shift s"
`<group>`	Groups multiple behavior codes under one button	`label`: Label appearing on interface for group of buttons
`<codeControls>`	Declares that codes are to be placed on the interface	`panel`: Specifies which side of screen code will appear on, "left" or "right" `label`: Specifies the label to appear on the interface for the set of behavior codes
`<global>`	Declares a global rating	`name`: Character identifier appearing in .casaa file `label`: Label appearing on interface `value`: Numeric identifier appearing

		in .casaa file `defaultRating`: Value on Likert scale that appears by default when a new file is coded `minRating`: Minimum value on likert scale `maxRating`: Maximum value on likert scale
`<slider>`	Places a declared global rating on the interface as a slider	`slider`: Name of global that is to appear (must match a name value in `<global>` code)
`<globalControls>`	Declares that global ratings are to be placed on the interface	`panel`: Specifies which side of screen global rating will appear on, "left" or "right" `label`: Specifies the label to appear on the interface for the set of global controls
`<row>`	Begins a new row when placing buttons	none
`<spacer/>`	Placeholder to leave empty space on the behavior coding or global ratings interface	none

General structure of userConfiguration.xml

The userConfiguration.xml file is organized as an XML file with all customizable components placed within XML tags. The XML file must begin with `<userConfiguration>` and end with `</userConfiguration>`. Comments may be made using standard HTML commenting syntax: `<!-- Example of a comment-->`.

Customizing behavior codes and subcodes

Behavioral codes must first be initially declared, after which the Graphical User Interface (GUI) must be setup to have the codes displayed as buttons.

Behavioral codes are declared within the `<codes> </codes>` XML tag. Each code is declared with the following elements:

- `name`: a unique string code name
- `value`: a unique integer value associated with that code

Example:

```
    <codes>
        <code name="CQ0" value="11"/>
        <code name="OQ0" value="14"/>
        <code name="SR-" value="18"/>
        <code name="SR0" value="19"/>
        <code name="SR+" value="20"/>
    </codes>
```

The above syntax declares five codes that can subsequently be setup to be displayed on the GUI.

Once each code has been declared, the user must establish how the buttons for these codes are to be displayed on the GUI. CACTI allows users to choose whether particular code buttons are to appear on the left or right panels, how these buttons are arranged in rows, and how multiple codes may be subsumed under one button (optional). The behavioral coding GUI is setup for each panel (left or right) using the `<codeControls> </codeControls>` XML tag. `<codeControls>` has the following elements:

- `panel`: must be "left" or "right" to indicate the setup for either panel
- `label`: specifies a label to appear on the GUI for each panel

Buttons are arranged in rows within the `<codeControls>` tag by using the `<rows> </rows>` XML tag, which delineates each row of codes. The `<button/>` XML tag is used between the `<rows> </rows>` tag to list each button that should appear in that row and specifies which code (declared above) the button corresponds to. The `<button/>` tag has the following elements:

- `code`: must be included for each button and corresponds with the code name declared above
- `key` (optional): allows users to select keyboard shortcuts for that button. Examples include "N", "shift N", and "shift ctrl N".

Multiple codes may be subsumed under a single button by placing multiple `<button/>` tags within a `<group> </group>` tag. For example, if a coding system specifies that certain behavior codes also must have a subcode for the valence or magnitude of that behavior code, it may be appropriate to place the subcodes within one common group.

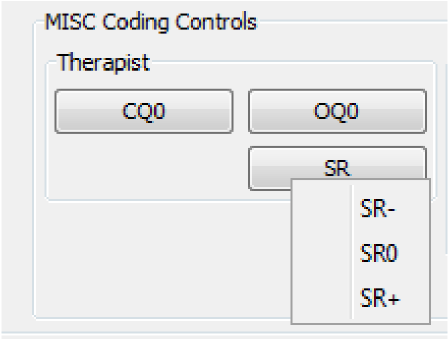
The `<spacer/>` tag will create blank space with an empty column in the current row.

Example:

```
<codeControls panel="left" label="Therapist">
```

```
        <row>
            <button code="CQ0" key="C"/>
            <button code="OQ0" key="O"/>
        </row>
        <row>
            <spacer/>  <!-- blank space on GUI -->
            <group label="SR">
                <button code="SR-" key="shift A"/>
                <button code="SR0" key="ctrl shift S"/>
                <button code="SR+" key="D"/>
            </group>
        </row>
</codeControls>
```

The above syntax specifies the setup for the left panel of the behavior coding GUI, which will be labeled as "Therapist", presumably because it contains buttons for therapist behavior codes. Buttons for the codes CQ0 and OQ0 will be displayed separately in the first row. Buttons for SR-, SR0, and SR+ will be displayed under one button in the next row labeled SR, which upon clicking, will allow users to select the appropriate SR-, SR0, or SR+ code. The buttons may also be activated by using the C, O, shift+A, ctrl+shift+S, or D keyboard shortcuts as specified.



Resulting code layout from above syntax

Customizing global ratings

Like behavior codes, global ratings must first be initially declared, then setup to be displayed on the global coding GUI.

Global ratings are declared within the `<globals> </globals>` XML tag. Each global rating is declared with the following elements:

- `name`: a unique string name for that global rating
- `label`: a label to appear on the GUI for the global
- `value`: a unique integer value corresponding with the global name
- `minRating` (optional): The minimum rating displayed on the Likert-type scale. The default is 1.

- `maxRating` (optional): The maximum rating displayed on the Likert-type scale. The default is 5.
- `defaultRating` (optional): The default rating assigned to this code. The default is 1.

Example:

```
<globals>
  <global name="ACC" label="Acceptance" value="0"
     defaultRating="3"/>
  <global name="EMP" label="Empathy" value="1"
     defaultRating="3"/>
  <global name="SEL" label="Self Exploration" value="6"
     defaultRating="1" minRating="1" maxRating="10"/>
</globals>
```

The above syntax declares three global rating variables, ACC, EMP, and SEL. ACC and EMP use the default 1 to 5 Likert-type scale, and SEL uses a 1 to 10 Likert-type scale.

Once each global rating has been declared, the user must establish how the sliders for these global ratings are to be displayed on the GUI. CACTI allows users to choose whether particular global ratings are to appear on the left or right panels and the order that they should appear in.

The `<globalControls> </globalControls>` tag allows users to place global ratings on the GUI. Users must specify a left or right panel for placing global-rating scales. Global-rating scales are placed using the <slider/> tag, specifying which global is to be placed using the global element, which must correspond with a global rating declared above. The `<spacer/>` tag may be placed to add blank space, manifested as an empty row in the current panel.

Example:

```
    <globalControls panel="left">
        <slider global="ACC"/>
        <slider global="EMP"/>
        <spacer/>  <!-- blank space on GUI -->
        <slider global="SEL"/>
    </globalControls>
```

The above syntax places three global ratings for ACC, EMP, and SEL in the left panel.

Parsing and coding an audio recording

Before an audio file is coded, it is required that a parser divides the audio file into mutually exclusive (but not necessarily exhaustive) units called utterances or parses. Parsing data, including beginning and ending times and byte numbers for each parse, are stored in a file with the extension .parse. Once parsed, parsing files are loaded through the CACTI and utterances are assigned behavior codes.

Parsing an audio file

To begin parsing an audio file, select Start a New Parse File from the Parse Utterances menu, and enter a name for the resulting .parse file that will be created, then select the audio file you wish to code. The .parse file will be created once the user enters the first complete utterance, and is updated automatically each time a new utterance is created.

The audio file must be in WAV format, not MP3 or other audio formats. This is mostly due to the variable number of bytes per second used in MP3 file structures. Because WAV files use a constant number of bytes per second, WAV file playback does not have this problem, and we strongly encourage only using WAV files with CACTI. Audacity (audacity.sourceforge.net) is a free and open source audio editor that will convert audio files from a variety of audio formats to WAV.

Press Play to begin audio playback. The cursor will begin to move on the audio timeline showing where the current playback is within the larger file, and the audio will play over the computer's speakers.

Press Start to indicate places when a new utterance begins and End to indicate where a parse ends. The placement of utterances will coincide with the timing of the audio playback when the Start or End buttons were pressed. Pressing Start multiple times in a row will automatically execute the End command, creating a sequence of utterances without gaps between them. Utterances will be displayed on the timeline to help with visualizing of where they have been placed. Utterances will appear in blue except for the most recent one, which will appear in green. Timing information for starting points of each utterance will appear on the timeline.

The CACTI parsing interface will display the timing information and byte numbers for the utterance that is currently being played, as well as for the utterance immediately before the current utterance.

A user may continue parsing an unfinished .parse file by selecting Resume Parsing a Parse File from the Parse Utterances menu.

Parsing Commands

- `Start`: Starts a new utterance. If pressed multiple times in a row, also ends an open utterance and starts a new utterance.
- `End`: Ends an open utterance. If pressed multiple times in a row, extends the end of the last utterance. Does not need to be used if a file is exhaustively parsed (i.e., without gaps between utterances.

- `Pause/Play`: Pauses or plays the loaded audio
- `Replay`: Backs up the audio playback to the beginning of the last utterance without changing any parsing information
- `Unparse`: Deletes the last utterance without changing the playback
- `Unparse & Replay`: Deletes the last utterance and moves the audio playback to the beginning of the deleted utterance
- `Rewind 5s`: Backs up the audio playback by five seconds without changing any parsing information

Coding an audio file

To code a file after it's been parsed, select Start Coding a Parse File from the Code Utterances menu, then select the .parse file you wish to code and a name for the .casaa file that will be created. The .casaa file will be created once the user enters the first behavior code, and is updated automatically each time a code is entered.

Press Play to begin audio playback.

Press any of the code buttons on the GUI to enter a code for the utterance that is currently playing. Codes may be entered in the middle of an utterance while the audio from the utterance is currently being played, or at the end of an utterance once it has been fully heard. If a user reaches the end of an utterance without entering a code, the audio playback will pause until a code is entered if the Pause if Uncoded box is checked.

If multiple codes are entered within a single utterance, the last code entered will be kept, thus a code may be changed by pressing the new coding option as long as the playback remains in the same utterance.

The CACTI coding interface will display the timing information, byte numbers, and codes entered for the utterance that is currently being played, as well as for the utterances immediately before and after the current utterance.

A user may continue coding an unfinished .casaa file by selecting Resume Coding a Casaa File from the Code Utterances menu.

Coding Commands

- `Pause/Play`: Pauses or plays the loaded audio
- `Replay`: Backs up the audio playback to the beginning of the current utterance without changing any coding information
- `Uncode`: Deletes the last code that was entered and moves the audio playback to the beginning of that utterance
- `Rewind 5s`: Backs up the audio playback by five seconds without changing any coding information
- `Pause if uncoded`: If checked, forces audio playback to stop at the end of an utterance if it has not been coded, requiring that all utterances have a behavior code entered. If unchecked, utterances may be left uncoded if the user does not enter a code before the end of the utterance.

- `Various Behavior Codes`: Behavior codes and subcodes that may be entered for each utterance. These may be changed by editing the userConfiguration.xml file, as described above.

Assigning global ratings

To assign global ratings for an audio file, select Score Global Ratings from the Global Ratings menu, then enter the file name of the .global file to be created and select the audio file to be associated with the global ratings.

Press play to begin the audio playback. Global rating variables with Likert response options may be adjusted by moving the sliders for each global rating. Only integer values may be assigned for global ratings. Global ratings are written to the .global file automatically.

Options

Options may be accessed under File > Options.

`Volume:` Volume adjustment is available in all interfaces (parsing, coding, and global ratings) and adjusts the volume level for audio playback.

`Balance:` Balance adjustment is available in all interfaces (parsing, coding, and global ratings) and adjusts the left-right volume balance for audio playback.

File structures

An explanation of the file structures for .parse, .casaa, and .global files are provided below. In addition, considerations that need to be accounted for during on-the-fly editing are also described.

Each of the file types can be opened, edited, and saved in Notepad or another simple text-editing program. Files are tab-delimited and therefore may also be read by spreadsheet programs such as Excel for data analysis; however, we recommend using Notepad or another simple text-editing program when editing CACTI files in order to best preserve their readability by CACTI.

The beginning line for each file indicates the location and file name for the audio file associated with the utterances or codes entered. The audio file location or name may be edited manually by changing this line:

```
Audio File:     C:\wave 1 sessions\Session12.wav
```

.parse file structure

Parsing data are stored in files with the extension ".parse". Each utterance is recorded sequentially on a separate line in the .parse file. For each utterance, the software records a sequential identifying number, the beginning and end clock times of each utterance, and the beginning and end byte numbers of each utterances, each separated by tabs.

```
[ID#  begin time  end time    begin byte  end byte]
```

```
0      0:00:01      0:00:07      208000       1264000
1      0:00:07      0:00:15      1264000      2752000
2      0:00:15      0:00:17      2752000      3088000
```

Note that the software reads the byte numbers, not the clock times, in determining where each utterance begins and ends with respect to the audio file. The clock times are used to demonstrate the length of the parse in a manner that is more interpretable to the user than the byte numbers.

If a user changes parsing information manually with simple text-editing programs, it is important to make sure that both the clock times and byte numbers are both changed to accurately reflect the desired change. For example, if a user wishes to combine utterances 1 and 2 above by directly editing the .parse file (note that this is also possible and easier to change within the GUI), they must do so by replacing the end time of utterance 1 with the end time of utterance 2, and by replacing the end byte of utterance 1 with the end byte of utterance 2, then delete the line for utterance 2:

```
ID#    begin time   end time     begin byte   end byte

0      0:00:01      0:00:07      208000       1264000
1      0:00:07      0:00:17      1264000      3088000
```

.casaa file structure

The .casaa file builds on a pre-existing .parse file by assigning a categorical language code to each utterance.

.casaa files contain records for utterances and behavior codes assigned to them. Each utterance is recorded sequentially on a separate line in the .casaa file. For each utterance, the software records a sequential identifying number, the beginning and end clock times for each utterance, the beginning and end byte numbers of each utterance, a number corresponding to the language code for that utterance, and the letter abbreviation for the behavior code, each separated by tabs.

```
[ID#   begin time   end time     begin byte   end byte     numcode char code]

0      0:00:01      0:00:07      208000       1264000      11     CQ0
1      0:00:07      0:00:15      1264000      2752000      44     FN
2      0:00:15      0:00:17      2752000      3088000      9      GI
```

Note that the software records two variables reflecting behavior codes, one a numeric code and one a string code. When changing the .casaa files by manually editing a code that was entered, it is important to make sure that both the numeric and string identifiers are changed together (note that it is also possible and easier to change these within the GUI). For example, if a user wishes to change utterance 1 to CQ0, they must change the string code from FN to CQ0, and change the numeric code from 44 to 11:

```
[ID#   begin time   end time     begin byte   end byte     numcode char code]

0      0:00:01      0:00:07      208000       1264000      11     CQ0
1      0:00:07      0:00:15      1264000      2752000      11     CQ0
2      0:00:15      0:00:17      2752000      3088000      9      GI
```

.global file structure

Data from global ratings are stored in files with the .global extension. Individual global rating names are indicated, followed by a tab and the Likert rating entered for that global rating:

```
ACC:  4
EMP:  4
SEL:  5
```

.backup files

Backup files are created automatically while parsing or coding behavior counts. These files have the same file name and extension as the .parse or .casaa files being parsed or coded, with an additional .backup extension. Backup files are duplicates of .parse or .casaa files that are currently being parsed or coded, except the data files are updated every 10 utterances rather than every single utterance. The purpose of these .backup files is to create a safeguard against data loss in the rare case that an error is incurred while parsing or coding. These .backup files may be saved or destroyed once a file is completely parsed or coded.

Data analysis

We have provided a basic routine to reformat sets of .casaa files to facilitate data analysis. The most current version of this MATLAB script, GetCASAA, may be downloaded from https://code.google.com/p/ctcsu-player/downloads/list. This routine will collect codes into an .SDS file suitable for lag-sequential analysis using the General Sequential Querier (GSEQ: http://www2.gsu.edu/~psyrab/gseq/index.html). Sample GSEQ commands are available from the same location. It will also produce a tab-delimited file with counts for each behavior code (coding variables in columns × recordings in rows), which can be imported into analysis programs including Excel, R, SAS, and SPSS. This routine may also be compatible with the open source MATLAB alternative Octave (https://www.gnu.org/software/octave/), although this has not been tested. Readers interested in further details about sequential analysis may wish to consult Roger Bakeman and Vicenç Quera's *Sequential Analysis and Observational Methods for the Behavioral Sciences*, published in 2011 by Cambridge University Press (ISBN: 978-0521171816).

Known issues

While coding an .mp3 file, the start and stop times of utterances do not correspond with the start and stop times that were entered when the file was parsed.

This issue is due to the variable number of bytes per second used within the .mp3 file structure, which causes start and stop times to not linearly map onto the corresponding byte numbers for start and stop times. While it is possible to parse and code MP3 files, it is recommended that files in MP3 or other formats are converted to WAV which will avoid this problem. Free programs like Audacity will convert most audio files to WAV.

Of particular concern for researchers with large amounts of audio data, WAV files are considerably larger than MP3 files. However, given that hardware for storing large volumes of data are becoming cheaper, larger in capacity, and smaller in physical size, this issue is becoming less of a concern. For example, in our lab, the WAV files we code are 10 times larger than MP3 files of the same duration would be, and a 50-minute WAV recording may require 500 MB of disk space (although the file size and playback quality can be increased or decreased by adjusting the number of bytes per second in WAV files). With files of this size, we are able to save 1000 50-minute recordings onto a single 500 GB external hard drive that can be purchased for well under $100.

This issue does not affect the assignment of global ratings to audio files; MP3 or WAV format is suitable for assigning global ratings.

Fitting buttons on screen

There is no limit to the number of codes or global ratings that may be used with CACTI. However, limited screen space may limit the number of codes or global ratings that can fit on the screen at once. Therefore, smaller computer screens may be particularly affected by this limitation. Typically, having too many behavior codes or global ratings will cause the CACTI GUI to be too tall to fit on the monitor. For behavior codes, this may be overcome by increasing the number of buttons per row (i.e., sacrificing height for width), or by using the <group> tag to subsume multiple behavior codes under one button, thereby decreasing the amount of space they take up. Spacing for global ratings may also be adjusted by ensuring that both the left and right panels have global ratings assigned to them; however, multiple global ratings cannot be subsumed under one button like behavior codes can. Therefore, in coding systems with large numbers of global ratings, CACTI may be unsuitable.