



LIPP

Logical International Phonetics Programs

Version 3.xx for Windows

Users' Manual

copyright 1990-2010



Intelligent Hearing Systems, Corp.
6860 SW 81st Street – Miami, Florida 33143
email: lipp@ihsys.com

Table of Contents

A. Introduction to LIPP	5
B. Getting LIPP installed and set up on your computer	8
(2) Installing LIPP	ALL 8
(3) Reminder about Placing the ACTIVATOR	UL,LL,SL 9
(1) Converting your keyboard to LIPP	ALL 10
C. A Tutorial for LIPP: How to get the system running	12
(1) Starting LIPP	ALL ... 12
(2) The "Settings" menu	ALL 12
(3) The Transcription module	ALL 14
(a) Entering the Transcription module.....	14
(b) Selecting a file while in the Transcription module	14
(c) Demographic Information	15
(d) Basic Transcription	15
(i) Typing on the "Gloss" Row	15
(ii) Lines and Rows in LIPP	15
(iii) Transcribing on the Target Form Row	15
(iv) Main Symbols and Diacritics.....	16
(v) Locations for diacritics	18
(vi) Transcribing on the Transcription Row	18
(vii) Lining up the target form and the transcription	18
(viii) Moving the cursor from Line to Line and Row to Row	19
(b) Transcription toggle features.....	19
(i) AUTO ADVANCE Toggle	20
(ii) AUTO REPLACE Toggle.....	20
(iii) Save Toggles	21
(c) Using the symbol menus.....	21
(i) The CONSONANT menu	21
(ii) VOWEL menu	22
(iii) Diacritics menu.....	23
(iv) Boundaries and global symbols	23
(4) File management in LIPP	ALL 24
(5) Using LIPP dictionaries	ALL 27
(a) Look Up Current Word.....	27
(b) Adding words to the dictionary	28
(c) Deleting words from the dictionary.....	28
(d) Undoing words selected from the dictionary	28
(f) Secondary indices	29
(6) Demographics and sessions in the Transcription	30
(a) Entering demographic and session information.....	30
(b) The meaning of demographic and session entries.....	32
(c) Saving demographic and session data.....	32
(d) Modifying demographic and session data	33

(e) Recalling demographic information for a new session	33
(7) Printing out your transcription files	ALL 34
(8) How LIPP understands phonetic symbols	ALL 35
(9) Analysis of Files in LIPP	UL/LL/SL37
(a) Different analysis types in LIPP	UL/LL/SL 37
(i) Segmental inventory analysis (context free)	37
(ii) Feature_counts_(context_sensitive)	UL/LL 38
(iii) Feature correctness analysis.....	38
(iv) Phonological Process analysis	39
(b) Performing an analysis in LIPP.	UL/LL/SL 39
(i) Displaying results of an analysis to the screen	40
(ii) Printing results of analysis	41
(iii) Saving results files.	UL/LL 41
(iv) Other analysis examples.....	41
(c) Reliability Analysis	42
(10) How LIPP analysis works	UL 44
(a) Definition of phonetic terms or variables in LIPP	45
(b) Writing rules in LAL	47
(c) Other analysis features	50
(d) A few quick tips on advanced LIPP rules	51
(e) Assimilations, metatheses, coalescences	52
(f) Exact length rules and recursive definitions.....	52
(g) Calculation metrics.....	52
(h) Phonetic avoidance analysis in LIPP	54
(i) Report Generation.....	55
(j) Multiple file analysis.	UL/LL 57
(11) Demographics in LIPP	ALL 59
 D. LIPP Symbology Development	 60
(1) Creating symbols in LIPP	UL 60
(a) Making a scratch alphabet file.....	60
(b) Entering "Edit Characters\"	61
(c) Viewing symbols.....	61
(f) Making and modifying symbols	62
(g) Clearing a symbol	62
(h) Exiting Edit Characters.....	62
(2) Keyboard mapping in LIPP	UL 62
(a) Removing symbols from the keyboard	63
(b) Stepping through the symbol list.....	64
(c) Placing symbols on the keyboard.....	64
(d) Fast movement in keyboard mapping	64
(e) Enabling symbols for FATTLIPP	65
(3) Printing new keycap covers	UL 65
(4) Redefining symbols	UL 65
(a) Stepping through the symbol list	66
(b) Stepping through the dimension values for a symbol	66
(c) Examining the meanings associated with the dimensions.....	66
(d) Changing symbol meanings	67
(e) Position adjustments	67

(f) Changing dimension meanings	67
(5) Selecting diacritic combinations	UL 68
(a) Changing diacritic combinations	69
(6) Adjustments in symbol menus	UL 69
(7) Defining new alphabets in general	UL 70
 E. Additional Features of LIPP	 72
(1) Creating Secondary Indices or Dictionaries	ALL 72
(2) Using the Special Transcript Line Features	ALL 72
(a) The tape counter feature	73
(b) The timing feature.....	73
(c) The space information feature	74
(d) Line comments.....	74
(3) File merge	UL/LL 74
(4) Save & Retrieve, Search & Replace	ALL 76
(5) Available alphabets	ALL 77
(6) Page saver printing: Print document compressed	ALL 78
 F. Index	 80
.....	61
.....	

1. Introduction to LIPP

LIPP stands for "Logical International Phonetics Programs". It was designed to make it possible to transcribe speech and speech-like sounds while working at an IBM-compatible computer. In addition, LIPP was created to make phonetic and phonological analysis of phonetic samples much easier and faster than doing it by hand and to make systematic analysis of transcription reliability possible. In the time since the release of LIPP 1.03, the program has been widely adopted and is now in use in many Universities and other facilities around the world that study phonetics and phonetic development. It is also used as a clinical tool by speech pathologists to transcribe and analyze speech of children with disorders of communication.

LIPP 2.xx for Windows represents a major revision of the originally DOS release. It includes several extensive new options and many improvements.

The theoretical phonetics and phonology that underlies LIPP is primarily the work of D. Kimbrough Oller, (that's Kim) a psycholinguist and phonetician, whose research for the past twenty years has focused on description of speech and speech-like vocalizations in infants and children, both normal and handicapped. The programming behind LIPP is the work of Rafael E. Delgado, a biomedical engineer, who has devoted much of his career to the implementation of computer algorithms intended to make life easier for people who are afflicted by speech and hearing handicaps. Kim and Rafael are the LIPP development duo.

Along with other Intelligent Hearing Systems (IHS) personnel, especially Ed Miskiel and Rebecca Eilers, the LIPP duo have created an easy way to do phonetic transcription, using real International Phonetic Alphabet (IPA) symbols in a system that includes many word-processing conveniences and special options that make transcribing both faster and easier. Working in LIPP, it's possible to be more attentive to transcription, because the computer system facilitates the process, provides clear screen displays and printouts of transcribed data, and reduces the time that it would otherwise take to analyze transcriptions.

There are actually three levels of LIPP. If you have THIN LIPP, you can enter transcriptions at your IBM keyboard (in a convenient manner with lots of word-processing conveniences) and print them out to any printer supported by Windows. If you have LOWER LIPP, you will have access to all the functions of THIN LIPP plus automatic phonological analysis using a library of analysis packages supplied by IHS (these include various phonetic inventory analyses and phonological process analyses). If you have UPPER LIPP, you will have access to all the functions of THIN LIPP and LOWER LIPP, plus the ability to modify the alphabets, keyboard placement of symbols, meanings of symbols and analysis rules you will utilize. UPPER LIPP offers the user a whole gamut of possibilities. Incidentally, if you have purchased THIN LIPP or LOWER LIPP, and find that you would like to upgrade your system to UPPER LIPP, everything you will have learned in working with the more basic versions of LIPP will apply to work with UPPER LIPP -- the systems have upward compatibility.

In addition to UPPER, LOWER and THIN LIPP special analysis packages, such as the Phonological Deviation Analysis by Computer (Hodson), may also be added.

Of course some users prefer a relatively simple system, one that is preset to a standard form of transcription and analysis, and for such users, two preset "standard alphabet" options are available: broad IPA and narrow IPA (we call them FATTLIPP and TITELIPP). Both these alphabets are discussed along with a rationale for choices of the various kinds of symbols in the section called "Available alphabets". However, some users prefer their own way of doing things, and UPPER LIPP allows them to have that flexibility. For example, if the user wants to create new symbols, that's easy in UPPER LIPP. If the user would like to place existing symbols in different places on the keyboard, that's also easy. The program even prints a new set of keycap covers for the new symbols the user might create. If the user wants to analyze data in a special way, it merely requires writing new analysis rules in LAL, the LIPP Analysis Language, which is very easy to learn, and is provided as part of the UPPER LIPP package.

LIPP is "logical" in several ways. First, LIPP keyboards are arranged logically so that users can usually find the symbols they want without any special learning. The "a" symbol is on the same key as the "a" on a standard computer keyboard, the "t" symbol is on the "t" key, and so on. For special symbols, the most commonly used ones are placed in prominent spots, and the less commonly used ones are placed in less prominent spots. All other things being equal, symbols that are related phonetically are clustered together on the keyboard.

A second way that LIPP is logical is that the analysis system allows definition of terms (for example, features such as stop, fricative, consonant, segment, etc.) using the simple logic of LAL (a kind of Boolean logic), and it also allows (in UPPER LIPP) the specification of new analysis procedures.

Third, LIPP is logical because underlying its symbology is a logically determined phonetic code that is the interpretive base of the analysis system. The code consists of a 16-dimension matrix that is predefined for THIN LIPP and LOWER LIPP users, and can be defined in UPPER LIPP however the user prefers. In the standard preset alphabets (TITELIPP, FATTLIPP) the 16 dimensions are predefined by the LIPP duo, and most users (especially THIN LIPP and LOWER LIPP users) will probably be satisfied with one of those sets of definitions.

Finally, LIPP is logical in that it is a truly "intelligent" system. It functions with enormous flexibility and is adaptable to many transcription needs. LIPP is sufficiently intelligent, that it can in many cases prevent the transcriber from entering an "illogical" transcription (thank you, Mr. Spock), for example a combination of symbols that doesn't make phonetic sense. For THIN LIPP, SPECIAL LIPP and LOWER LIPP users, the illogic traps are built in according to Kim's specifications. But in UPPER LIPP even the logic of the combinations can be specified by individual users.

The Quick Reference Guide for LIPP 1.40 can be used as a schematic to help the user find LIPP options. Section A will direct you on how to set up your keyboard and install LIPP. Section B

presents a tutorial to help you get started in LIPP. If you haven't had LIPP training, you should be sure to read this section and follow the instructions in order to learn how to be an efficient transcriber. LIPP can help you be a much faster transcriber, but you need to know how to use its conveniences in order to take advantage of its speed. A well-schooled user of LIPP will be a fast and accurate user. The tutorial sections are marked to indicate whether they are relevant for THIN LIPP, SPECIAL LIPP, LOWER LIPP or UPPER LIPP. "ALL" next to a section title means the section is relevant to all four levels of LIPP. Otherwise there will be an indication of UL, meaning the section is relevant to UPPER LIPP only, or UL/LL, meaning it is relevant to both UPPER LIPP and LOWER LIPP, or UL/LL/SL meaning it applies to all but THIN LIPP.

After the tutorial, the remainder of the manual includes more detailed information on LIPP conceptualizations and functions.

2 - Getting LIPP installed and set up on your computer:

2.1 - Installing LIPP

ALL

Before installing LIPP, if you have purchased an UpperLIPP or LowerLIPP version of LIPP, insert your SuperPro ACTIVATOR into the first parallel port of your computer (where you plug in your printer). If you intend to use your printer, plug it into the other end of the activator. Make sure it's all the way in, and that the little screws on each side are set so that it doesn't slip out.

Note: As of 2004, Activators are no longer needed to run LIPP. Each version of LIPP is registered to a specific user and the name of that user appears on the bottom of the main LIPP module. That user is responsible for any unlicensed copies distributed to other users.

In addition, before you install LIPP, you should check to make sure you have enough disk space to hold the program and run it. You need quite a little space -- about two megabytes to run everything easily. If you try to run some LIPP function and it doesn't work, it may well be that you simply don't have enough disk space to accommodate that function. Also remember that if you begin to transcribe lots of files and store them on your hard disk, you may use up the space LIPP needs to run. So it makes sense to keep track of your disk space, always leaving a few megabytes to absorb new files you may create.

To install LIPP, simply insert the LIPP CD in your CD ROM drive. The installation program will automatically start running. If for some reason it doesn't start running, simply search for the SETUP.EXE program on the CD using Windows Explorer. Double click on the SETUP.EXE program to start the installation.

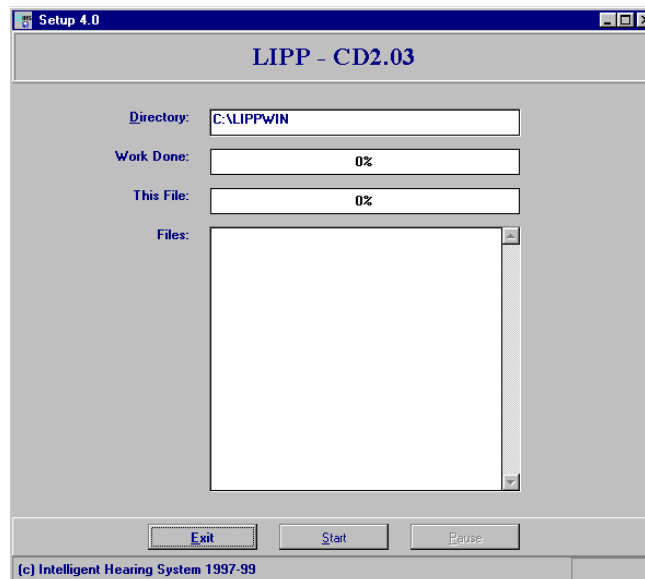


Figure 2.1 – Installation program. Press [Start] button to initiate the installation of LIPP on your hard drive. You may change the name of the directory where LIPP will be installed.

Now follow the instructions on the screen as they appear, and the installation program will generate a "LIPPWIN" directory and copy all the LIPP program files to your computer. It will also install the two alphabets (FATTLIPP and TITELIPP) with their dictionaries and a number of example files of transcripts and analyses. (If you aren't going to use both alphabets, you can delete the one you don't use later):

The installation program will automatically setup a program group called "LIPP" in your Program options in the Start button (lower left side of screen). Inside the program group you will find a "LIPP Program" option. Select this option to start running LIPP.

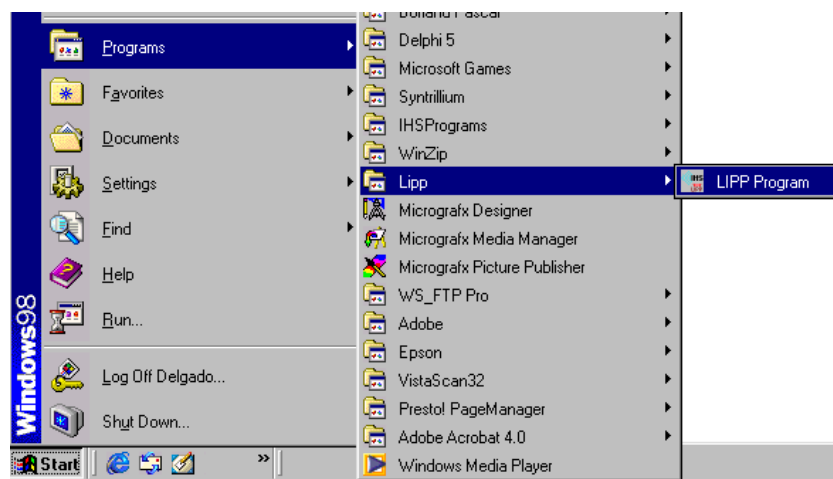


Figure 2.2 – Starting your LIPP program from the Start Menu.

Now you can remove the LIPP CD and store it someplace safe just in case something goes wrong with your hard disk. If you ever do have problems with the hard disk and can't run programs, you will need to get it fixed and then reinstall LIPP.

2.2 - Reminder about Placing the ACTIVATOR UL,LL,SL

LIPP WILL NOT RUN WITHOUT THE ACTIVATOR CLIP INSTALLED ON YOUR FIRST PARALLEL PORT (WHERE THE PRINTER CABLE GOES), UNLESS YOU HAVE THIN LIPP.

Note: As of 2004, Activators are no longer needed to run LIPP. Each version of LIPP is registered to a specific user and the name of that user appears on the bottom of the main LIPP module. That user is responsible for any unlicensed copies distributed to other users.

2.3 - Converting your keyboard to LIPP

ALL

(If your keyboard has already been converted, skip this section, and go to "Installing LIPP"). Your computer keyboard has caps that are labeled with standard keyboard symbols. For phonetic transcription, you need phonetic symbols, so you will probably wish to adapt your keyboard for LIPP. IHS sells a variety of keycaps for different types of keyboards and even a fully adapted, ready-to-plug-in keyboard. If you have purchased a set of LIPP-adapted keycaps, you may wish to begin by installing them.

If you don't want to put on new keycaps, you can skip the remainder of this section and go to "Installing LIPP". But if you don't adapt the keyboard, you may need to use a LIPP keyboard diagram (see Appendix 1) in order to know where each of the symbols is.

Along with the keycaps will have come a printed sheet with all the symbols of the TITELIPP alphabet. If you plan to install keycap covers for an alphabet other than TITELIPP, you may wish to print out the keycap covers for your selected alphabet (see section "Printing new keycaps"). FATTLIPP, is a broad transcription alphabet, and its symbols practically are all included in the narrower alphabet, TITELIPP, so most users who work in FATTLIPP do not feel it necessary to redo the keycap covers that come with the LIPP manual -- by using the TITELIPP keycap covers, they have access to all the symbols they require, and in addition can see on the keycaps additional symbols that are available in the narrower alphabet.

You should find the TITELIPP keycap sheet in Appendix 1 of the manual. The symbols are printed on spaces designed as keycap inserts. First cut out the inserts as indicated by the dotted lines. (With non-IBM keycaps you will need to cut the inserts smaller than indicated in order for them to fit into the transparent portion of the caps). Then pop the keycaps off one at a time (it's easy on an IBM keyboard -- just pull gently on the key and it will come off, exposing a smaller blank key below -- it's a little harder with other kinds of keyboards, but the keycaps can be popped off without damaging the keyboard if you do it carefully), and replace them with the transparent keycaps including the appropriate inserts arranged so that the phonetic LIPP symbols (in blue and red if you are using color coded keycap covers) are on top of the keys when installed, and the regular computer keyboard symbols (in black if you are using color coded symbols) are on the front faces of keys. Be very sure you get the inserts in the right places, so that the front faces of each key when installed show the same symbols that were on the standard computer keycaps they replaced. If you replace keys one at a time, it's hard to make mistakes.

Since the LIPP keys have standard symbols on their front faces when you finish making the replacements, you don't lose access to your keyboard for nonphonetic programs or other purposes. When you use your computer for something other than transcription in the future, you will merely look at the front faces of the keys to determine what key to strike. When using LIPP, however, you will look at the top of the keys. You'll notice that you don't have to replace all the keycaps, because some of the IBM keys are used without any change in LIPP. When you've replaced all the keys that were provided, put the standard keycaps that you just removed into the package and put it in a safe

place in case you want to switch keyboards around at a later date.

If you have decided to use special symbols you wish to create yourself, you will need to go through a series of steps described later in the manual (starting at the section called "LIPP Symbology Development"). When you follow the procedure indicated there, you will develop a keyboard designed to your own specifications with keycaps that are appropriate, but unless you have a color printer, you'll have to use black and white only for the new keycaps.

Although converting a keyboard is fairly easy, some users prefer to have a special LIPP keyboard that can be plugged in whenever their computer is being used for phonetics and phonology work, along with another regular keyboard that is plugged in at other times. If you would like to obtain a special LIPP keyboard, it can be purchased through IHS.

2.4 - A Tutorial for LIPP: How to get the system running:

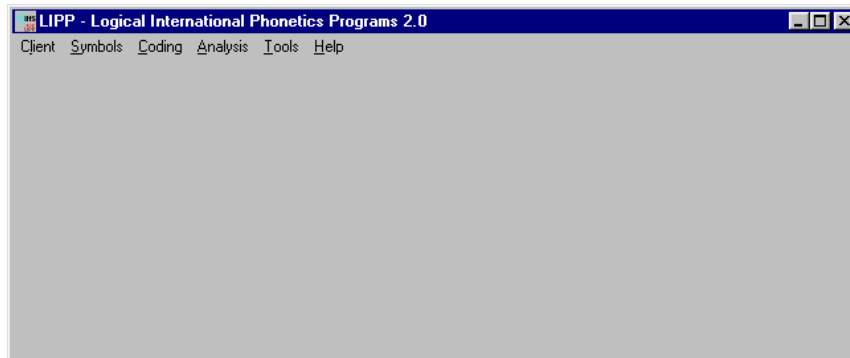


Figure 2.3 – LIPP Main Screen

2.4.1 - Starting LIPP

ALL

If you plan to use your printer, you should make sure it is connected and turned on before you start running LIPP. If you start running LIPP without the printer being on line, you may not be able to use the printer until you reboot the computer again. The printer port will be locked up until you reboot. So if you want to use the printer, turn it on before you start running LIPP.

Now to run LIPP, go to the Start button and look for the Programs option. Now look for the LIPP program group and select the “LIPP Program” option. See **Figure 2.2** above.

The program will begin running, and after a few moments, you will see the colorful Main menu appear. By using the mouse (or keyboard – see your windows manual for selecting menu items using the key board. Note that all menu items have an underscored letter that is used to activate the menu item using the keyboard) you can select what you want to do.

2.4.2 - The "Symbols" menu

ALL

The "Symbols" menu will tell you that one of two alphabets is in place (TITELIPP or FATTLIPP) and will have an indication about the printer that will be available in your work with LIPP. If you select "Alphabet", you will see the alphabet options available on your disk (depending on which one or ones you installed, FATTLIPP or TITELIPP or perhaps several others). If you installed both alphabets, try this out, switching the settings back and forth between TITELIPP and FATTLIPP, but then leave it on the alphabet you want to use when you're finished.

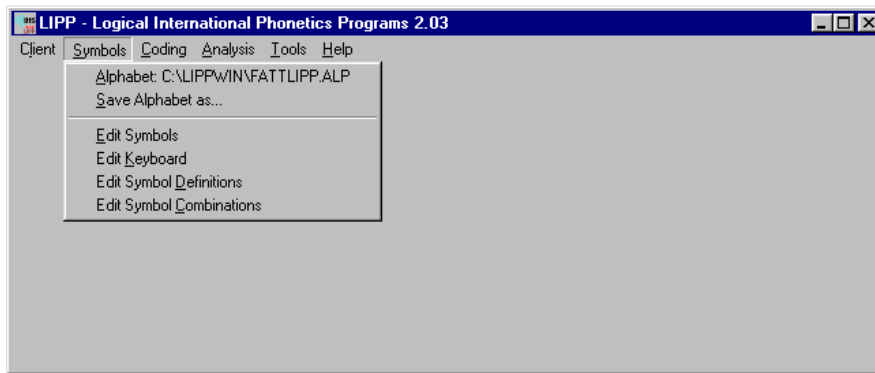


Figure 2.4 – Symbols Menu. The first item in this menu indicated the alphabet you are currently using. Always make sure it is set to the correct alphabet before starting a transcription.

When you begin loading previously existing transcription files, you may at some point see a message that says "Warning", and cautions that the alphabet used in the original transcription is not the one that is currently selected under "Symbols". This warning is important when you are using widely divergent alphabets, but on the whole FATTLIPP and TITELIPP are highly compatible (FATTLIPP being largely a subset of TITELIPP). Consequently, you can almost always ignore this compatibility warning and proceed (press the [OK] button to load file under the currently selected alphabet) as long as the two alphabets are FATTLIPP and TITELIPP. Otherwise you will need to Exit the Transcription module, go to "Symbols", select the appropriate alphabet, and then return to the Transcription module.

Other options in the "Symbols" menu will allow you to save the current alphabet under another name (use this option in order to generate a new alphabet using the current alphabet as a starting point), edit symbols, edit symbol definitions, edit symbol diacritic combinations and symbol keyboard locations.

3 - The Transcription module

ALL

You will probably want to follow along the next sections in order. We're going to get into transcription now, away from the Main menu, so once you have entered the Transcription module and you're ready to quit for the day, you will need to go to the "File" menu, and select the "Exit LIPP Transcription" option (last item in the "File" menu).

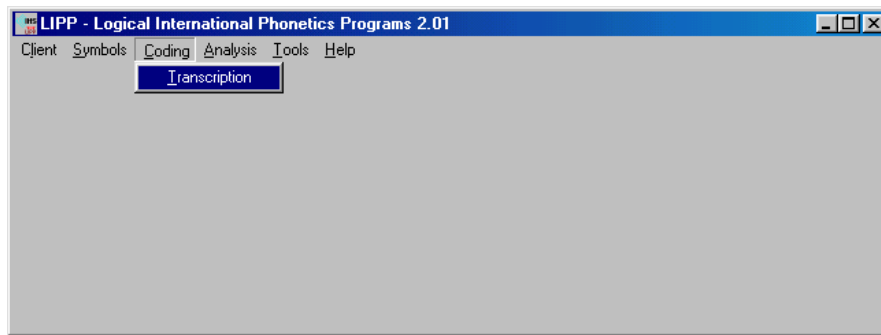


Figure 3.1 – Selecting the Transcription Option from the main menu.

3.1 - Entering the Transcription module:

Let's go to Transcription now. Press the "Transcription" option in the "Code" menu of the Main module. See **Figure 3.1** above.

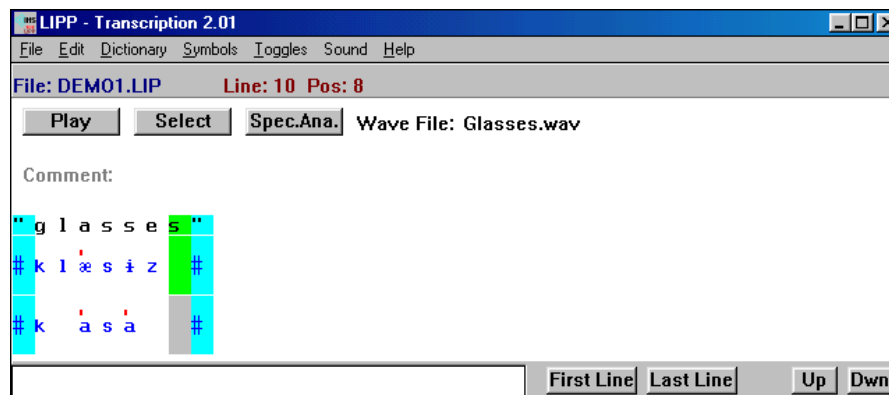


Figure 3.2 – Transcription window.

3.2 - Selecting a file while in the Transcription module:

The first thing that happens when you enter Transcription is that you will be presented with the options to continue with the last transcription file that was used on the computer. If you wish to continue with that file, press the [YES] button and LIPP will automatically load the file. If you do not want to continue, press [NO]. LIPP will allow you to enter the transcription module to start a new file or load, from the "File" menu a different file.

3.3 - Demographic Information:

Almost everyone who does transcription intends for any file they create to include a header that indicates who the speaker is, what the date of the session is, and perhaps certain other demographic information. Whenever you start a new transcription file, or begin to work with a transcription file, you should enter the demographic information selecting the database option in the [File] menu in the transcription module.

3.4 - Basic Transcription:

You should now see the screen configured for transcription (if not, read again above at a. Entering the Transcription module). The gray cursor will be on highest of the three Rows. The top row is the **Gloss** row used to enter standard text. The second row is the **Target** that is used to enter to “target” pronunciation of the words being transcribed. The lower line is used to enter the actual transcription. Notice that the cursor appears in green over the corresponding symbols in the rows not being edited.

3.4.1 - Typing on the "Gloss" Row. When you are on the "Gloss" Row, your keyboard acts just like a standard keyboard, and what you type appears on that Row with standard symbols. Type the following:

"happy"

Be sure to type the quotation marks as well as the letters. Notice that LIPP automatically inserts word boundary symbols (#'s) underneath the quotation marks on the two transcription Rows. If you make mistakes while typing, you can enlist standard word-processing conveniences, such as moving the cursor to the Position of the mistake and striking the <Delete> key, the <Insert> key (which will provide an empty space while advancing all characters to the right of it), or the <Backspace> key (which will delete mistakes to the immediate left of the cursor). Try it out. Change some of the letters, then retype the correct spelling of "happy", and be sure to keep the quotation marks. If you accidentally move below the "Gloss" Row, you can get back to it by hitting the <up arrow> key.

3.4.2 - Lines and Rows in LIPP. LIPP transcription files are composed of "Lines". Notice at the top of the screen that there are indicators of the location of the cursor in terms of "Line" and "Position" (abbreviated "Pos" on the screen, this term means "column", of which LIPP offers 40). Each Line has three "Rows", a Gloss Row, a Target Form Row, and a Transcription Row. You should be on Line 1 at this point. Press the <Page Down> key, and you will move to Line 2. Notice that what you typed in the Gloss Row disappears, because now you are on a new Line. Press the <Page Up> key, and you will return to Line 1. What you typed on the Gloss Row reappears. You can also move down Lines by using the <Enter> key, which will advance your cursor by one Row each time it is struck. When it is on the Transcription Row, and you strike <Enter>, you will advance to the next Line. To come back up to Line 1, again use the <Page Up> key.

3.4.3 - Transcribing on the Target Form Row. Now position the cursor on the Target Form Row under the word "happy" using the arrow keys. As soon as your red cursor is on the Target Form Row, you have access to the phonetic alphabet you selected. Move to the first open space

between the # symbols and type
h ae p i

The [h] is just where you would expect it to be, on the key that has "h" on a standard keyboard. The [ae] is not on standard keyboards, but it's a very commonly used phonetic symbol, so it appears in an unshifted position in all the standard LIPP alphabets, on the key that has the number 6 (just above T and Y). The numerical keys over on the right hand side of the keyboard are used for other purposes in the standard LIPP alphabets, so if you press the "6" over there, you won't get [ae].

While you're still on the Target Form Row, try moving the cursor around using the arrow keys, deleting and inserting some of the other symbols. Again if you get off the "Target Form" Row, you can return to it by using the "up or down arrow" keys, or if you get on the wrong Line, you can return using <Page Up> or <Page Down>.

Notice that in standard LIPP alphabets the symbols are arranged in a "logical" fashion on the keyboard -- the phonetic symbols that look like nonphonetic symbols are in the same places the standard keyboard puts them, the most commonly used symbols are in prominent spots, and other symbols are organized to be clustered near phonetically related symbols. The largest symbol printed on each keycap is the one you get when you press the key unshifted. The top symbol among the smaller printed symbols on each keycap is the one you get if you press the key while you hold down the <Shift> key. The second one down is the one you get if you press with <Ctrl> (or <Control>), and the third if you press with <Alt>. The LIPP duo has tried to assign the unshifted values in the TITELIPP and FATTLIPP alphabets to the most commonly used symbols for transcribing English and Spanish, since those are the languages we have so far tended to work with the most.

3.5 - Main Symbols and Diacritics:

To do phonetic transcription in LIPP, it's important to understand the difference between Main Symbols (sometimes also called "Base Symbols") and Diacritics. Main Symbols are in blue, and Diacritics in red on our color-coded keyboards. A Main Symbol goes in the center of a space (i.e., a particular Position on a particular Line). A Main Symbol represents a phonetic segment, like a [b], a [p], or an [ae], or it represents a boundary marker of some sort (like a word boundary, "#"). Diacritics go in spots around the periphery of a Main Symbol. Diacritics are used to modify the detailed meaning of Main Symbols. For example, if you type the symbol [p], you can add the Diacritic superscript "h" by typing <Shift> "h". The combination of [p] and superscript "h" denotes an aspirated labial stop consonant.

Note that you do not have to move the cursor back to the Main Symbol in order to add a Diacritic, as long as the cursor is resting on the open space immediately to the right of the Main Symbol. LIPP assumes you want the Diacritic to go on the Main Symbol you just typed, and the program moves the cursor back, puts the Diacritic where it belongs, and then advances the cursor again. This procedure is very convenient when you are typing fast and adding lots of Diacritics. Of course, if you wish, you can simply use the arrow keys to position the cursor directly over any Main Symbol, and then add a Diacritic.

LIPP will not allow you to type a Diacritic before you type a Main Symbol for it to modify. You need to have the cursor either in the empty space just to the right of an appropriate Main Symbol, or you need to have it directly over the appropriate Main Symbol. Try to type a Diacritic by positioning your cursor with the "right arrow" key in a blank space, several spaces to the right of your transcription, and hitting <Shift> "h". You should get an error message at the bottom of the screen indicating that you typed the Diacritic before you typed a Main Symbol. Of course an unshifted "h" can be put in the Main Symbol Position, but it doesn't have the same meaning in LIPP as the shifted "h" -- as a Main Symbol, it represents a consonant. Also try typing the "colon" symbol, at <Shift> "semicolon" (just to the right of the "L" key), while the cursor is at an open space, several spaces to the right of your transcribed symbols. Again, LIPP tells you that this is not permitted, since the colon symbol is a Diacritic, not a Main Symbol. Now type [a], and before moving the cursor, type <Shift> semicolon, to get the colon-like IPA symbol that means "lengthened". An [a] with a colon after it means a "long [a]".

LIPP will also prevent you from typing many nonsensical transcriptions. For example, suppose you type [p], and then <Shift> F9, to get the little square symbol that IPA and standard LIPP alphabets use to indicate a "laminal" or "tongue blade" articulation since the 1989 revision of the IPA. Well, [p] does not involve a tongue articulation at all, so LIPP tells you (try it) that, this Diacritic doesn't go with this Main Symbol in the TITELIPP alphabet. On the other hand, if you type [t], and add <Shift> F9, you'll see a symbol combination that means a [t] with laminal or "blade" articulation.

Most of the Diacritics used in the narrow standard LIPP alphabet, TITELIPP, are found on the FUNCTION keys (in most keyboards they are at the very top, or over on the left hand side), in <Shift>, <Ctrl>, and <Alt> positions. The unshifted FUNCTION keys are used for menu functions, as we shall see later. If you accidentally press an unshifted FUNCTION key (these keys are typically reserved for Windows functions and cannot typically be used for applications), and end up in an unfamiliar place, you can usually get back to the transcription module by pressing the right mouse button over the transcription module form. Try out some of the Diacritics on the other FUNCTION keys in order to get familiarized with use of Diacritics. For example, the combination [t] and <Shift> F4 (unreleased) means an unreleased [t].

If you delete a Main Symbol, you will delete its Diacritics as well, and then you must start over at that Position. Notice that you can put more than one Diacritic on a Main Symbol. Type [p], and then press <Alt> F9. This will give you a little house-like symbol under the [p] that means "labiodental". Now add the <Shift> "h" Diacritic that means "aspirated". Now look at the result, a [p] with a little house under it, and a little "h" above it and to the right, denoting an aspirated, labiodental [p]. Now type [p] again, in an open space, and add the labiodental symbol. Then type the colon from the same key that gives the colon from a standard keyboard (<Shift> semicolon). This yields the special colon-like symbol that means "long" in the IPA. Now add <Shift> "h", and you'll see that the superscript "h" gets put much higher above the [p] than in the case where the colon was not used. That's because the superscript "h" and the colon prefer to occupy the same Diacritic location, the one to the right of the Main Symbol.

3.6 - Locations for diacritics:

There are five possible locations for Diacritics in LIPP (top, top-right, right, bottom-right, and bottom). Only one Diacritic can go in each location. So, when the colon is already to the right of the Main Symbol, the superscript "h" has to take a secondary position, top-right, which makes it go quite high on the screen. But its meaning in LIPP is the same as if it were in the preferred position.

3.7 - Transcribing on the Transcription Row:

Now that you have had a chance to play a little with the symbols, adjust the Target Form Row so that it says

h ae p i

and hit <Enter>. Now you are on the Transcription Row, the bottom one on the screen. Let's suppose that the child you are listening to says the word "happy" with no "h" and with the wrong vowels. We'll try to enter a transcription for the child that shows just what he says. Type the following in the spaces underneath the Target Form.

a p a

The symbols should be entered in such a way that the [p]'s line up on the two Rows. Also the [a]'s on the Transcription Row should line up with the vowels on the Target Form Row. This way LIPP will know that you, the transcriber, believe that the child has deleted the target [h] and has substituted [a]'s for both vowels. When you're finished, the three Rows should look like this

Gloss Row:

" h a p p y "

Target Form Row:

h ae p i

Transcription Row:

a p a

3.8 - Lining up the target form and the transcription:

Because of the way LIPP analyses work, it's important that everything on the Target Form Row and the Transcription Row (including the boundary symbols) line up properly. This means that Transcription segments that correspond to Target Form segments should be placed in the same "Position" or column as the Target Form segments. For example, if the Target Form has a consonant that the speaker deletes, be sure that there is a blank space on the Transcription Row just below the target consonant that was deleted. If the speaker adds an element (by what phonologists call "epenthesis", for example), be sure to put a blank space on the Target Row just above the speaker's element that is not present in the target. Sometimes, lining up requires difficult decisions since it is not clear to which target elements the speaker's units correspond. Special analysis options can help account for such special errors, but the transcriber is still well-advised to try to maintain alignment of each Target Form and Transcription as much as possible. The decisions you make in lining up are phonetic and phonological ones, and they will determine to some extent the level of insight your analysis can have. LIPP does not require you to line up any aspect of the Gloss Row with the Target

Form or Transcription, but you will find it easier to interpret your transcripts if at least you keep the boundary markers on each Row aligned.

The above example is properly lined up. The following one is not.

Gloss Row:

" h a p p y "

Target Form Row:

h ae p i

Transcription Row:

a p a

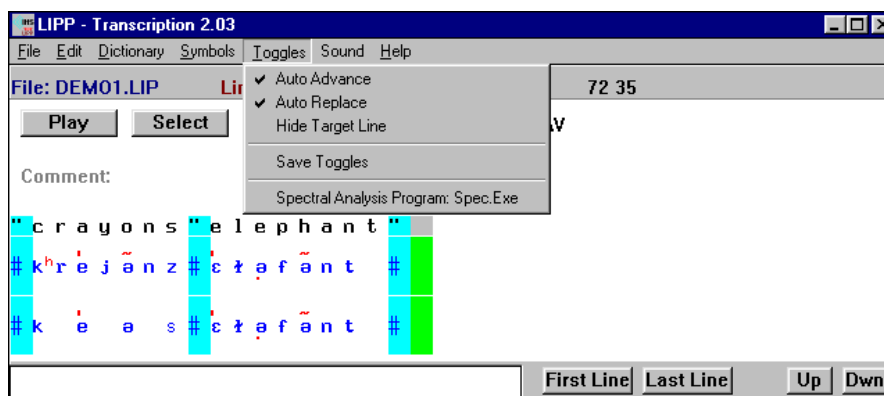
3.9 - Moving the cursor from Line to Line and Row to Row:

If you get off the Line you're working on, you can return to it using the <Page Up> and <Page Down> keys. Press <Page Down> now to go to a new Line (Line 2). Since Line 2 is empty, you should end up on the Gloss Row, Position 1, the first available Position on that Line. Whenever the cursor is moved to an empty Line, LIPP takes it to the first Position of the first Gloss Row. On the other hand, if the Line is already filled, <Page Down> will take the cursor to the same Row (Gloss, Target Form, or Transcription) at the new Line that the cursor was at on the original Line. The cursor will automatically go to the second "Position" (i.e., column) in that Row (unless the first is unfilled). This positioning has substantial advantages if you are using a standard transcript file (for example, words from a standard assessment instrument) where all the Gloss Rows and Target Form Rows of each Line are already present in a preset file and your task in each transcription is merely to enter or make modifications in a Transcription Row. In such a case, it is best for LIPP to provide a means by which you can easily move to the Position at which you are most likely to wish to begin your transcribing. Since the first Position will (in standard LIPP files) have a boundary symbol, the second Position is the one to which the <Page Down> key moves the cursor. We'll see an example of this kind of movement of the cursor later.

The <Enter> key can also be used to move down Lines. If you begin at a Transcription Row, and press <Enter>, you will move down a Line, but unlike with the <Page Down> movement, the cursor will be positioned on the Gloss Row, Position 1. If you press <Enter> again, you'll move down exactly one Row, with the cursor going to the first Position of the next Row. The <Enter> key is set to allow Row by Row movement, while the <Page Down> key is set to allow Line by Line movement in the most convenient way possible. "Up arrow" and "down arrow" keys can also be used to move up and down within Lines. You move the cursor up Lines by using <Page Up> (remember you can refer to the "Line" indicator at the top of the screen, in order to know where you are in the transcription file).

3.10 - Transcription toggle features:

In the Toggles menu there are a number of "toggle" features that allow the transcriber to set certain characteristics of function in ways that suit the immediate task.



3.10.1 - AUTO ADVANCE Toggle. In many cases the LIPP transcriber may be working with a file where Transcription Rows are empty at the beginning of the session. In such cases, the transcriber will use relatively few Diacritics and may wish to have the cursor advance every time he strikes a symbol key. This is the way LIPP is normally arranged. On the other hand, if the transcriber is using lots of Diacritics, it may be preferable to have the cursor stay put until the space bar is struck. That way the user can put on several Diacritics before moving ahead to the next symbol location, and not have to use the "back arrow" key repeatedly to get back to the location of the symbol where additional Diacritics may be desired.

To prevent the cursor from moving forward automatically, you need to toggle "off" the "AUTO ADVANCE" feature in the Toggle Menu. If you haven't done it already, go to the Toggle menu and click to the AUTO ADVANCE toggle. Notice that the feature's changes from "checked" (meaning that the feature is ON) to "not checked" (meaning that the feature is not ON). Note that the check mark will appear on the left side next to the menu item. Type a [p] and then <Shift> h. With the AUTO ADVANCE feature off, the cursor stays at [p] until the superscript "h" is added. Now hit the <space> bar to advance the cursor.

Before going any further in the tutorial, return to the Toggle menu and reset the AUTO ADVANCE toggle to "checked".

3.10.2 - AUTO REPLACE Toggle. LIPP offers other toggle features that allow the user to select a mode of function that suits his/her fancy at the moment. An especially useful one is the AUTO REPLACE toggle which allows you to make revisions in an existing transcript without having to use <Delete> and <Insert> to create an open space before changing a symbol.

By default, the AUTO REPLACE Toggle is "Checked". You can check this by going to the Toggle Menu and scrolling to the AUTO REPLACE toggle. Click on the Auto Replace toggle to check and uncheck the feature. Now return to the Transcription Row. Position your cursor over the [p] and type a [t]. If AUTO REPLACE is "checked", the [p] and any Diacritics that accompanied it, will disappear and be replaced by the [t]. You didn't have to hit <Delete> and <Insert> to replace the symbol. If the AUTO REPLACE Toggle is "unchecked", you will be presented with an error

message.

AUTO REPLACE is a useful toggle because sometimes you want to revise a previous transcript, rather than creating a new one from scratch. When AUTO REPLACE is "checked", you can do revisions quickly. It's also useful when you are using a word list from a standard assesment instrument that you have prepared ahead of time. In that case you don't have to create a new transcript entirely, because the Transcription Rows of the test will (if you wish it so) already be filled with transcripts of correct pronunciations. Your task in administering the test is merely to enter changes in the Transcription Rows so that they match what the child (or other client) actually said. For this task, you may wish to have AUTO REPLACE "checked".

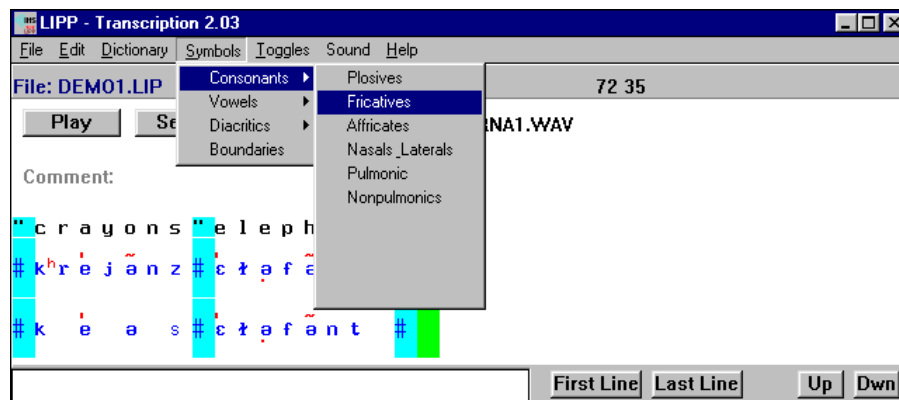
One hint: AUTO REPLACE works only with Main Symbols. You cannot AUTO REPLACE with any symbol that can function as a Diacritic.

If your AUTO REPLACE is now "unchecked", go back to the Toggles Menu and turn it to "checked".

3.10.3 - Save Toggles. We'll talk about the other toggle options later. If you have a consistent preference on toggle values (different from the ones Rafael put into LIPP as "default" values), you may wish to select the values you want and then "save" them, by using the "Save Toggles" feature in the Toggles menu. That way each time you start running LIPP, it will begin with the toggle values you selected, instead of the ones Rafael put in as defaults.

3.11 - Using the symbols menus:

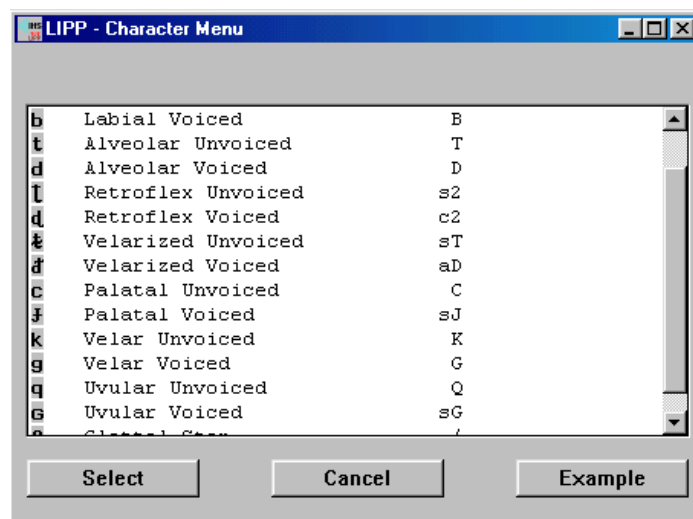
If you have trouble finding symbols that you want to use, it's easy to get to them using the SYMBOL Menu. Now make sure your transcription cursor is on either the "Target Form" Row or the "Transcription" Row, because otherwise you cannot access the symbol menus (if you are on the Gloss Row, LIPP phonetic symbols are not available). You might need to <Esc> and use the arrow keys to move the cursor to an acceptable Row. Once your cursor is in the right place, you can access any of the three symbol menus.



3.11.1 - The CONSONANT submenu (Symbols Menu). The CONSONANT submenu is

organized by feature grouping. Notice by looking at the top row of the menu, that the plosives (or stops) come first, then fricatives, then affricates, etc. You may also notice that there are lots of symbols in the Consonant menu, perhaps some that do not occur in the FATTLIPP alphabet. The LIPP duo has elected to leave all the symbols in the TITELIPP alphabet in all the preset menus, even though many of them are suppressed from the keyboard in FATTLIPP. If you have chosen to use FATTLIPP (or some other broad alphabet) and ever change your mind and want to access any of the narrower symbols not in your selected alphabet, you can enable the narrower symbols by simply mapping them onto the keyboard (see the section called "keyboard mapping" under "LIPP Symbology Development") yourself quite easily, as long as the symbols you want are in the menus.

Now select the symbol for a voiced bilabial fricative, the Greek letter "beta" (it looks like a capital b with a tail on the left side). While this symbol is highlighted, click the left mouse button. Notice that the menu disappears, and (assuming your transcription cursor was over a blank space) the symbol for the voiced bilabial fricative now appears on the Target Form or Transcription Row (wherever your cursor was at the time). You have just selected a particular symbol from the Consonant menu.



3.11.2 - VOWEL submenu (Symbols Menu). Let's go to the VOWEL menu. When the VOWEL menu is selected, you will see that there are two available feature groupings: rounded vowels and unrounded vowels, and you can access the one you want. Select the vowel [ae]. Again the menu disappears and the [ae] appears on the Transcription Row.

Go back to the VOWEL menu, and let's look at the way it's organized. In the first column of each feature grouping is a symbol, just as it will appear when you type it. In the second column is a description of the meaning of the symbol, written out orthographically. In the third column is an indication of where on the keyboard the symbol can be found. In the third column, a lower case "s" means <Shift>, a lower case "c" means <Ctrl>, and a lower case "a" means <Alt>. So in some cases, you may wish to use the menu in order to find where a symbol is on the keyboard, but you

may not wish to have it appear on your Transcription Row when you exit the menu.

Let's try that. Search the VOWEL menu using the arrow keys until you find "open o", the one that looks like a backwards c, designated as Mid Back Lax under rounded vowels. Notice that the "open o" is indicated to be at keyboard position "sO", meaning <Shift> "O". Now hit <Esc> and you will return to the Transcription Row. Hit <Shift> "O" and the "open o" will appear on your Transcription Row.

3.11.3 - Diacritics menu. Now place your cursor over the "open o" on the Transcription Row, and go to the Diacritics submenu (Symbols Menu). If you are using FATTLIPP, some of the Diacritics of a narrow alphabet will be (as long as wish it that way) unavailable to you, even though they appear in the menu. However, all the standard LIPP alphabets include a Diacritic for voicelessness. It consists of a little circle that IPA prefers to put underneath the Main Symbol that the circle modifies. Find it by using the down arrow key on the first column in the Diacritics menu; it will be next to the word "Voiceless". When you have found the little "voicelessness" circle, you will see that it occurs on the same key that has a sort of backwards apostrophe (the unshifted value) in IBM standard keyboards. The "nasalization" symbol is also on that key usually in the upper left hand corner of the keyboard, next to "tab". To get the "voicelessness" symbol, you will need to hit <Shift> "backwards apostrophe (nasalization)", or else select the symbol in the menu and then hit <Enter>. When you do it, you'll see the little circle appear underneath the "open o". The combination symbol you have now created denotes a voiceless or whispered vowel. Remember that you can't put a Diacritic in an open space, so you have to select a vowel first, place the cursor over the vowel and then put your Diacritic on.

3.11.4 - Boundaries and global symbols. The Symbols Menu and the Diacritic submenu also includes columns for boundary symbols and some special symbols that are helpful if you want to transcribe super broadly. Suppose, for example, you are transcribing an utterance that seems so garbled you can't really tell what consonants and vowels it has, but you can tell that the utterance consists of a CVCV sequence. If you wish, you can transcribe it just exactly that way. Go to the Diacritics menu and look under "Global Symbols" for capital "C" and capital "V" (scroll down to find them with the "Base Symbols" or "Main Symbols". You'll see that they are on the LIPP keyboard just where you might expect them to be, at <Shift> "C" and <Shift> "V".

The symbols available for global transcription and boundary marking within the IHS supplied alphabets offer some special capabilities for transcription. In the section called "Available alphabets", the characteristics of the alphabets supplied by IHS are discussed in some detail.

Go back to the Transcription Row with an <Esc> and type in a CVCV sequence. Notice that although these symbols (along with the ones for "unspecified nucleus", "unspecified syllable" and "quasivowel") appear in the Diacritics menu, they are actually Main Symbols, just one of our little quirks.

4 - File management in LIPP

ALL

Now that you know how to find symbols and type them, it's time to learn how to find transcription files and save them. Go to the File Menu and select the Open option. LIPP will ask you if you want to save the file you were creating during the last period. Say "no", press the [No] button. Then LIPP will go get the list of available transcription files (those with a .LIP extension in their filenames). The LIPP duo supplies you with DEMO files for instructional purposes, and a list of them should appear now. Select DEMO1.LIP and click on the [OK] button. In a moment the DEMO1 file will be found for you, and the first Line of the file will appear on your screen. (If you get an alphabet warning, it means that under the Symbols menu, in the Main LIPP module, an alphabet is selected that is different from the one that was used to transcribe DEMO1; don't worry as long as the two alphabets in question are TITELIPP and FATTLIPP, just hit the [OK] button, because the two alphabets are highly compatible).

Notice, by looking at the top of the screen, that you are at Line number 1. Using the [Page Down] key, go down one utterance (i.e., one Line) at a time, and look at the various utterances that have been transcribed. Note that the [Page Down] key takes you automatically to the second Position on the Row from which you originated as it changes Lines. You will see that at Line 3, the gloss is "black" and the Transcription indicates a deleted [l] and a [k] that has been changed to a glottal stop. Note that every present segment is lined up from Target Form Row to Transcription Row, and that deletions are also properly aligned. If no gloss and target form are entered, it indicates that the transcriber did not know the meaning of the transcribed utterance, or that the utterance was "babbling"; page down to Line 8 and you will see such an utterance.

You can make revisions in the transcript of the file if you like, using all the methods you learned above. Try out a few of those possibilities, deleting symbols, inserting spaces, changing symbols. If you want you can go to the last Line of the file (press the [Last Line] button on the bottom right side of the transcription module screen area, and look at the Line number), and then you can transcribe some additional Lines.

When you're satisfied you know how to move around in the file, then save the new file you've created. Remember that you are working on DEMO1.LIP, but you can save it under a new name, and that way it won't change the DEMO1.LIP file at all -- it will still be in the memory as before. Here's how. Go to the File Menu and select the Save option to save the file under its current name or Save As... to save the file under a different name. If you selected the Save as... option, LIPP will display the a Save File Dialog Box to allow you to enter the new name while displaying the current files with the LIP extension. Make sure to use a filename that is composed of not more than eight letters and/or digits plus a .LIP extension. For example, you might use the name MYFIRST.LIP. Then hit <Enter> and your file will be saved under that name. Don't worry if you made the mistake of saving the new file under the name DEMO1.LIP, because you still have a copy of the original DEMO1.LIP on the installation CD, in case you want to restore it.

All transcription files should have an .LIP extension, or else they won't appear in the menu you get when you select "Open" option from the File menu. You can select a different directory folder by clicking on the displayed folders in the Open Dialog Box.

5.1 Using standard speech-sound assessment files:

Now that you know how to read and save files, consider for a moment one of the ways you may want to use LIPP in transcribing of pronunciations from standard speech sound assessment instruments, or laboratory phonetics tests you use repeatedly. You may have purchased the Phonological Deviation Analysis by computer (Hodson) package, which includes the Hodson assessment procedures file HODENG.LIP (or HODSPA.LIP for the Spanish version of the test). Or you may wish to create a word file of your own to use in phonological evaluations, for the sake of argument let's call it TEST1.LIP, that has all the glosses and all the targets you want to examine. When you transcribe the subject's (or client's) pronunciations of the test items, you merely read in TEST1.LIP and then go through the items one by one, transcribing without having to put in glosses or targets. This is the same method you would use with the Hodson procedure.

NOW BE CAREFUL! When you save the file you have created by editing your assessment instrument file, you should give the edited version a special name, for example JohnTest.LIP (perhaps meaning a test of a client named John), rather than using the name of the original test file. If you simply select the Save option rather than the Save As... option, you will be naming the edited file with the name of your test file, and the original test file will no longer exist. **THUS WHEN YOU TRANSCRIBE IN A TEST FILE, ALWAYS USE THE "SAVE AS..." OPTION TO SAVE IT UNDER A NEW NAME.**

You may want to think about a system for naming files that will suit your purposes. Everyone seems to want to do that their own way, but it does pay to have a method predetermined so that when you go looking for files in directories, you know how to find them based on their names. If you don't have a preference about creating file names, then try this advice: use six characters from the subject's name or number (if you assign numbers to your subjects) plus a two digit session number and then the .LIP extension.

LIPP has a special convenience to speed your work with preset tests. As you move the cursor from Line to Line using the [Page Down] key, you will advance automatically to the second Position of the Row from which the cursor originated. Thus if you are editing Transcription Rows only, you can position the cursor on the Transcription Row of Line 1, make your transcription or modification of transcription, and then hit [Page Down] to move the cursor to the second Position of the Transcription Row on Line 2. This, the LIPP duo have discovered, is the Position most transcribers will want to start transcribing from on most lines. The <Page Down> setting puts you in a place that minimizes the amount of cursor movement you have to do with arrow keys and <Enter>s.

4.2 Inserting and deleting Lines:

Suppose that when you give your test, you call up a file containing all the relevant words, in the right order, but the child ends up pronouncing some of the words, more or fewer times than you expected. Instead of creating a whole new file for this situation, you can revise the file as you go by selecting the “Delete” and “Insert” options in Edit Menu. These options automatically adjust the numbering for all the items in the file.

To delete, simply position yourself at the Line that you wish to delete, select the Delete option. The current Line will disappear and the next line will appear in its place. If you use the [Page Up] and [Page Down] option you will see that the line is no longer present in the file. To insert a line, simply position yourself at the Line above which you wish to have the new Line appear, and select “Insert” option, again from the Edit menu. A new blank Line will appear where you can insert the utterance you wish to (perhaps because you forgot to transcribe it while going through your tape the first time).

5 - Using LIPP dictionaries

ALL

For those of you that don't use a standard test, it may be helpful to use LIPP dictionaries while you transcribe. The FATTLIPP.DIC dictionary (and its companion, the TITELIPP.DIC dictionary), compiled by Michele Steffens, and adjusted by Kim and Rafael, has lots of words in it, with target forms included. This is how to use dictionaries.

5.1 Look Up Current Word:

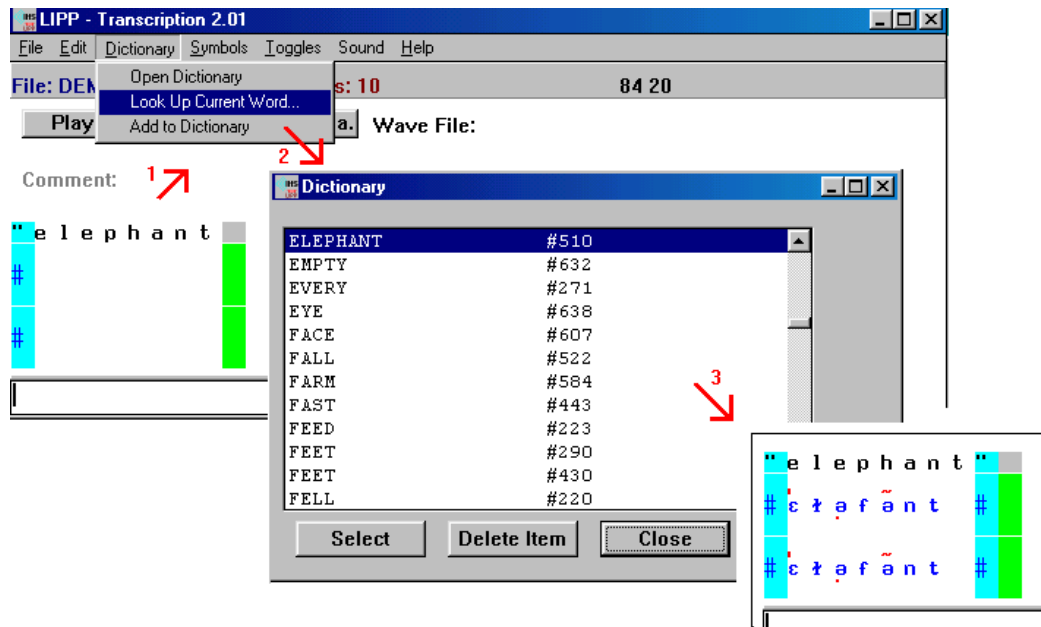


Figure 5.1 – Looking up words from the dictionary is easy – simply start typing the word and then select the “Look Up Current Word...” option in Dictionary Menu. Select desired word to add to transcription document.

Position your cursor on a Gloss Row and type quotation marks. Now type the first two letters (actually you can type all the letters if you like) of a word that is on a test you like to use, for example, "ele", for "elephant". Now go to the “Look Up Current Word” in the Dictionary menu. LIPP will now search the dictionary to find the first word (in alphabetical order) that begins with "ho". The dictionary dialog box will appear with the highlighted option that comes closest to the entered word. If that word is "elephant", then "elephant" will be highlighted. To copy that word to your transcription, simply click on the [Select] button at the bottom of the dictionary dialog box. If there are other "ele" words ahead of "elephant" you need to scroll down using the mouse to click up or down on the scroll bars. When "house" is highlighted, then you can click on [Select] to copy the word to your transcription. By checking on the Fill Vocal option, you can tell the LIPP to fill both the transcription and target rows (checked) with the selected word or to only fill the target row (unchecked). Depending on the type of transcription you are working on, it is sometimes easier to

have the dictionary fill both rows and then simply edit the transcription row to match the actual vocalizations.

One advantage of this sort of dictionary access is that you can have some control over the target forms you use. If you enter targets new on each Line, you'll run the risk of making small changes in the targets, and this may be a bother at the time of analysis.

Another advantage of the LIPP dictionary approach is that for most words (especially long ones), it is much faster to look them up then to type the whole target.

If you get into the dictionary, and you don't find the word you want, or you want for any reason to go back to transcription, just click on the [Close] button. You can also use the "Open Dictionary" option on the Dictionary Menu to simply look at all the words in the dictionary. In that case, you can scroll through the alphabetic list by clicking on the scroll bar or pressing the up/down arrow keys until you find the word you want.

5.2 Adding words to the dictionary:

If you want to add a new word to the dictionary, it's easy. Go to the Gloss Row, type your quotation marks, then the word you want to add, then quotation marks; then go to the Target Form Row, type in the transcription you want to use as a target for the gloss you indicated, then position the cursor on any of the three Rows between the boundary symbols for the word you want to add. Now select the "Add to Dictionary" option in the Dictionary Menu. The word between the word boundaries will automatically be added to the dictionary. To verify that the word has been added, simply press the "Look Up Current Word" option and you will see the new word in the list.

If you try to add a word that is already in the dictionary, LIPP will accept the word, filing it in the dictionary adjacent to (above) the one that's already there, on the assumption that you may wish to store more than one target form for a particular gloss.

5.3 Deleting words from the dictionary:

If you don't like the target form provided by Kim and Michele, you can change it by using the "Delete Item" option from the Dictionary dialog box. You will first need to use the Open Dictionary or Look Up Current Word option to access the Dictionary Dialog Box. If you use the Look Up Current Word option, use the steps described above to enter to word you wish the delete in the Gloss row so that the dictionary will open on or near the word you wish to delete. When the Dictionary dialog box opens, select the word you want to delete and click on the [Delete Item] button. This will delete to selected word and will automatically disappear from the list.

5.4 Undoing words selected from the dictionary:

If you select a word from a dictionary and install it on a transcription Line, then realize that

you have made a mistake, you can remove the word by using the delete keys to remove the word.

The FATTLIPP dictionary is the biggest one IHS offers to users, and it can be accessed easily. The difference between the FATTLIPP and TITELIPP dictionaries is minimal, and has to do with some symbology assumptions more than anything else. In TITELIPP we transcribe initial English phonemically "voiced" stops with voiceless unaspirated segments (to indicate the way they are usually pronounced in American English) and we use the IPA "upside down r" to indicate the English retroflex semivowel. In FATTLIPP we use the rightside up "r" for English "r" and we use "voiced" stops in initial positions (because they are often pronounced voiced and it simplifies certain analyses to have them treated as phonetically voiced).

The following feature was available in DOS, but is not currently available in the Windows version:

(f) Secondary indices: *"Secondary indices" are special dictionaries where the words are not alphabetized, but are listed in an order you specify, with group titles you also specify. This dictionary approach may be useful if you want a categorization of words you wish to use in an examination of a child, for example, and you want those words to be grouped by some phonetic characteristic, say the number of syllables in the word, or the final consonant of the word, or the vowel quality in the stressed syllable of the word. A "secondary index" must have a .SDC extension. See the section called "Secondary Indices" for further discussion of this option.*

6 - Demographics and sessions in the Transcription Module:

ALL

Now it's time to take a look at the Demographics functions that can be accessed within the Transcription module. We designed LIPP demographics this way, in order to prevent users from losing track of transcription data that they might otherwise forget to label. Experience has taught us that this kind of misplacement can happen as soon as you have a dozen files or so. Therefore we recommend that you enter demographic information for all your files and that you add the files to the session log of each subject.

The screenshot shows a software window titled "LIPP - Demographics". At the top, it displays "Client Number: 2" and a text field for "Name:" containing "John Smith". To the right of the name field are two buttons: "Load Client" and "New Client". Below this is a tabbed interface with three tabs: "Demographics", "Address", and "Session". The "Demographics" tab is selected and contains the following fields: "ID:" with the value "12345678901"; "Birth Date:" with sub-fields for "Month" (3), "Day" (29), and "Year" (1998); "Gestation:" with the value "35" and the unit "Weeks"; "Current Age: 2.6 years"; "Language:" with the value "English"; "Disability:" with the value "None"; and "Sex:" with radio buttons for "Male" (selected) and "Female". A checkbox labeled "Corrected" is checked. At the bottom of the dialog is an "OK" button.

Figure 6.1 – Demographics entry dialog box accessible from Transcription Module (File Menu) or from Main Module (Client Menu).

6.1 Entering demographic and session information:

If you are not in the Transcription module, enter it, and open the Demo2.LIP file. When the file comes up, go to the File Menu and select the database option to enter demographic information.

The cursor will be on "Name". If the file corresponds to a new subject, click on the [New Client] button to assign a new database number to that subject before entering any information. Each subject or client has a unique identifier. You can also load information corresponding to a previously entered subject by clicking on the [Load Client] button and selecting the appropriate client from the list that will appear. Note that the demographics dialog box will open displaying the demographic information for the last subject entered or to the subject associated with the file that was loaded.

To enter information for a new subject, click on [New Client] button and type a name (for example, Bill Howard). You can also use the mouse to select the various fields where you want to enter data. The demographics dialog box contains three tabs for information corresponding to general demographics, address and sessions. You can use the mouse to open each of the tabs and enter the corresponding information. For the subject ID, you can enter any alphanumeric value (up to 20 characters). For example, you can enter 2A333XQ for the ID. You can also enter the subject's birth date by filling in the month, day and year options in the demographics tab. The month value must be entered as a numeric value from 1-12 (1 or 01 will be understood as January, 2 or 02 as February, and so on). A two-digit "year" value will be understood as corresponding to the 1900's (92 will be read as 1992, 76 as 1976, and so on). For birth years corresponding to the year 2000 or higher, enter 2000, 2001 etc. In general, we recommend that you enter the full birth year to avoid confusion. When you finish, the birth date of Bill Howard (or whatever name you entered) will be calculated in days, months or years based on the date in your computer and will be displayed in the current age display in the demographics tab. Always make sure that your computer has the correct date and time.

The "gestation" slot is for entering information about pre-term infants in weeks. Forty weeks is normally considered full term, so 36 would mean 4 weeks pre-term. You can enter data on gestational age if you wish, or you can skip this option if your work does not involve reference to gestational age. LIPP will automatically (within a module called Generate Multi File List) provide file lists that are arranged by gestationally corrected ages if you chose to utilize the "gestation" slot to code prematurity. As with the "gestation" slot, you may enter "address", "phone", "language", "disability" and "sex" information if you wish, but it is not obligatory.

The Session tab will show all that files that have been transcribed for a particular client. You must update this information when you transcribe a file. The best way to do this is to enter the demographics module from the transcription module. Then you can select the subject you are working on (or add a new subject to the database), enter the name of the file, comment, transcription date and the transcriber's name. Then press [Add File] button. The file will appear on the list and the corresponding information transferred to the file header. LIPP will also automatically calculate the age of the subject at the transcription date based on the dates you entered for both the transcription and the birth date.

The Demographics module allows you to enter information for more than one category within slots (except the numerically coded date or age slots). So for example, you may have a client who speaks English, Spanish, and Finnish, in which case you might choose to put all three languages in the "Language" slot. The syntax of such an entry merely requires that you place a comma between the entries, and LIPP will understand. Thus if you choose to abbreviate languages to their first three letters, then

Eng, Spa, Fin

as an entry under "Language" could mean that the client is from a home where all three languages

are spoken.

6.2 The meaning of demographic and session entries:

In fact, all the slots in the Demographics screen (aside from dates) will have whatever meanings you want them to have (although you cannot change the slot titles). LIPP provides you a schema that can be used as it is or else you can define it for your own purposes. Thus "ID" can be a local identification number of the subject, and for most users that's the way that slot will be used, but if you wish, it can be another kind of name, or it can be a coding for some feature of the child, such as his/her hair length. It can be whatever you chose it to be, but you should set up a method that is systematic so that whatever features you enter in each demographics slot can be referenced for database purposes (and we'll talk about that later under "Multiple file analysis" and "Generating file lists"). So if hair length is important to you, and you categorize data in terms of hair length in inches, then be sure every entry of hair length in demographics, is provided in inches.

Because the demographics slots are not strictly defined, it may be to your advantage to define some of them a special way. So if "language" is always "English" in your work, or you don't really care if other languages are involved in the subject's life, then the "language" slot can be subverted to other purposes. It should be noted, however, that some of the slots (address and phone) are unavailable to referencing by the LIPP database system, so if you want to use a slot for special categorizations not indicated in the Demographics screen, choose a slot other than one of the "address" slots or the "phone" slot.

6.3 Saving demographic and session data:

Now that you have entered data in the Demographics screen for Demo2.LIP, you are in a position to save information in three ways. First of all, if you save the transcription file at this point, the demographics entries you have made will become a header on the Demo2.LIP file. The next time you call up the file and enter the demographic dialog box, you will see the data corresponding to the subject displayed.

The second way you can save the demographics information is to enter it in a permanent client database on your computer. This can be accomplished by selecting the [New Client] button. The database number assigned to the client is merely a number to indicate uniquely the order in which this client was entered into the database. It is sometimes useful to know that number because it will allow you to call up all the data you have on that client very quickly. And it will allow you to start work with a new session on a previous client without having to enter the demographic information. When you select the [Load Client] button, a dialog box will appear with all the names of the clients in the database. Note that the clients are in alphabetical order as entered in the name field. If you want them to be alphabetized by last name, make sure you enter the last name first.

The third way to save information in the Demographics screen concerns the individual session. If you choose the "Add current file to database" option at the bottom of the Demographics

screen, the transcription filename (in this case Demo2.LIP) will be stored in a session file that is referenced to the client information. This way, if you transcribe another session from the same client later, you can keep track of both of those sessions (in fact any such sessions) and keep them all associated with the appropriate client data. If you select the "Add current file to database" option, you will be queried as to whether or not you truly wish to add it. Don't do it unless you really want to, because it creates a file that is hard to delete. If you do add it, you will see a schematic of its information appear at the top of the screen. Each time you add a new session for the client, it will be added to that list. You can return to the Demographics screen with <Esc>. To look at the list again, select "View sessions in database"; when finished, hit <Esc>.

The name of database files is determined by the name of the alphabet plus an extension that is .DMG or .SES. If you change alphabets, the demographics and sessions information for the files you transcribed using the previous alphabet will not be referenced to the database you previously established. You may need to make adjustments by renaming or copying the .DMG or .SES files. The important point is always to remember that changing alphabets can produce problems unless you think through all the implications of doing it. Thus if you have selected FATTLIPP under "Settings", the client database into which the demographic and sessions information will go, will be in a file named (by the program) FATTLIPP.DMG, and the sessions information in a file named FATTLIPP.SES.

6.4 - Modifying demographic and session data. Any time you are in the Demographics screen (within the Transcription module), you can modify the demographics data or sessions data for the particular session that the current file represents. However, in order to make the modifications available to database functions, you must make the same modifications in another location (see below in the section on the "Demographics module"). If you try to use "add new client to database" or "add current file to database", LIPP will understand you to want to put new, not modified client or session information in the client database. By saving the file with modified demographics, the header for the file will be adjusted, but it will not be referenced properly to the database even if you select "add new client to database", or "add current file to database". In order to make such modifications you need to go into the Main menu and select Demographics to enter the main Demographics module (see below), and make the changes there.

6.5 - Recalling demographic information for a new session. When one begins to transcribe a new session with a previous client, all that is necessary to obtain the data on demographics for that client is to select the [Load Client] button and a list of all the currently entered clients will appear. Merely select the client from the list and all the demographic information for that client will automatically appear.

7 - Printing out your transcription files ALL

To printout your transcription file, you will of course need to have an appropriate printer installed with your IBM computer. Any printer that is supported by Windows will work with LIPP.

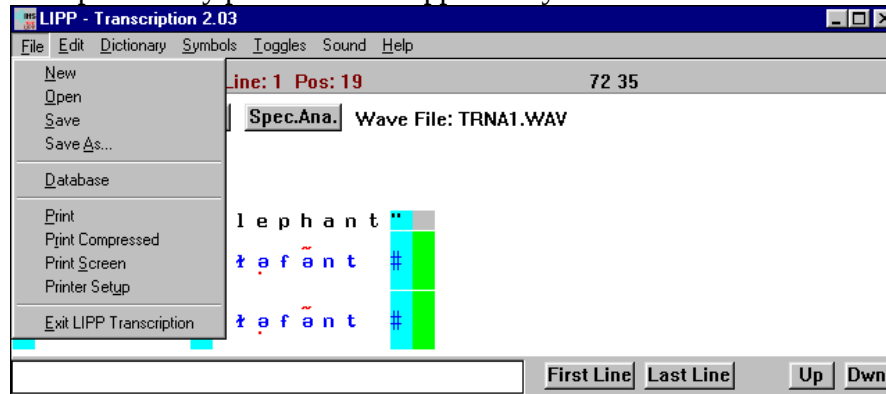


Figure 7.1 – File menu in the Transcription Module showing print options.

Let's try it if you have your printer hooked up. Go into the Transcription module, and read the file DEMO1.LIP (if you're not already in it). Now go to the File menu and select the Print option. The current file will be printed. In addition to the standard printing options, you also have a Print Compress option that will print as many transcription lines as possible on a single print line, a Print Screen option that will generate a print-screen of the transcription window and the Printer Setup.

In addition to these printing options, you may also grab the current screen and insert it as a bit map image into a Word document by pressing the Print Screen button on your keyboard and then pasting on to your document.

8 - How LIPP understands phonetic symbols (ALL)

LIPP knows what each phonetic symbol means because when it performs an analysis it interprets each symbol as a 16 dimension value that is stored in a big table in LIPP's memory. Each dimension represents a phonetic feature, and each symbol can have any of 20 values on each dimension, so there are many thousands of possible meanings that LIPP can accommodate. Actually, TITELIPP only has about 200 symbols, and the narrower alphabets have fewer. The maximum number of symbols that can be used in the present form of LIPP is 240.

The symbol [p], for example, represents a voiceless, unaspirated, stop consonant. LIPP knows that, along with a variety of additional details. The code for [p] includes a "1" on the first, or "A" dimension, the dimension that refers to "degree of jaw closure and/or airflow" in standard LIPP alphabets. The "1" means that [p] is a plosive (closed jaw, no airflow until it's released). The second or "B" dimension shows a "0" for [p] because that dimension refers to "tongue frontness", a feature that is unspecified for [p], since the tongue can be in various positions while articulating a [p]. The "C" dimension refers to "lip posture", and [p] gets a "1" on that one, meaning it is a "bilabial". The "D" dimension is "veloglottal function", where [p] gets a "0" because it is neither a nasal, nor a click, nor any of several other possibilities. The fifth or "E" dimension is "glottal state and voicing" and there [p] gets a "1" for being "unvoiced and unaspirated". We won't go through all the rest of the feature dimensions here, but suffice it to say that [p] gets a "0" on most of them because it is what linguists call "unmarked", meaning it doesn't have any other special characteristics.

Now the symbol [m] is interpreted quite differently. It gets a "7" on the "A" dimension (to indicate its nasal airflow), then a "0" (for unspecified tongue position), then a "1" (for bilabial), then another "1" (for lowered velum), and a "2" on the "E" dimension (for voicing).

Each symbol has its own meaning in the 16 dimension code. The global symbols like C and V also have meanings in the code. They get "0's" on almost everything, because we leave them unspecified on all the features except for the ones that differentiate consonants and vowels.

Main Symbols have a full characterization on all 16 dimensions. Diacritics on the other hand are merely modifying symbols, so they have specification only on certain dimensions, that modify the meanings of Main Symbols. For example, the diacritic "superscript h" which means "aspirated" has no effect on its Main Symbol except on the "E" dimension, where it changes the value to a "3", meaning "unvoiced and aspirated".

Appendix 2 indicates the coding system for TITELIPP as presently formulated, and Appendix 3 is a table with all the codes for the some 200 symbols specified. You can printout a table like the one in Appendix 3 for whatever alphabet is selected under "Settings" simply by selecting the Characters module in the MAIN Menu and selecting "Print Character Definitions".

Some of the global symbols you see in the Diacritics menu are really Diacritics. For example,

suppose you wish to indicate that a particular sound is an affricate consonant, but you don't wish to make any additional claims. Type <Shift> "C" to get the "unspecified consonant" symbol, then place your cursor over the "C" and type <Shift> "A", to produce a superscript capital "A". LIPP understands that combination to mean "affricate consonant, otherwise unspecified". The transcriber who types this combination is making no claims about voicing or place of articulation.

Boundary symbols, such as the word boundary (#), are also accessible from the Diacritics menus. The boundaries (on the "tab" key) are interpreted by LIPP to be non-segments with a special coding (with a non-0 value on the "N" dimension). Blank spaces on the Target Form or Transcription Row are also understood by LIPP, although they simply get a "0" on every dimension.

9 - Analysis of Files in LIPP (UL/LL/SL)

Phonological analysis in LIPP can be conducted in a number of different ways. First we'll go through some examples, and that may be all you want to hear on the topic. But if you want to see how to write your own LAL rules, or wish to understand how they function, then we'll tell you how the analyses are performed and how you can create custom analyses if you want them.

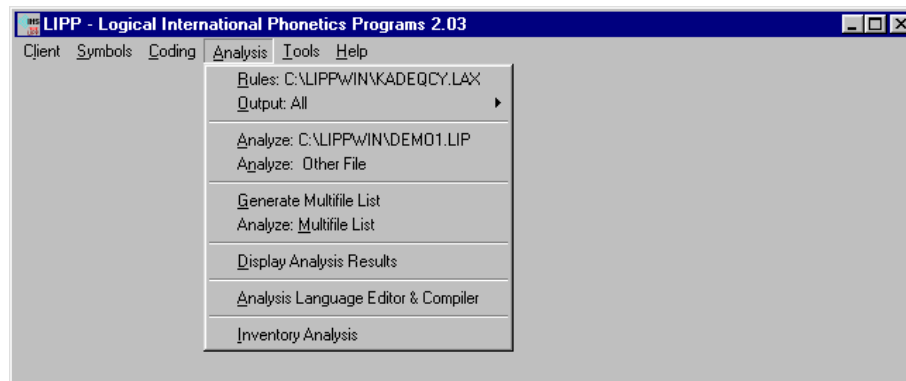


Figure 9.1 – Analysis Menu

9.1- Different analysis types in LIPP: (UL/LL/SL)

9.1.1- Segmental inventory analysis (context free). The simplest kind of analysis in LIPP is (context free) segmental inventory correspondence analysis, which will determine automatically not just what segments were produced by the client (in your transcript), but also what target sounds they were based on (Transcription <- Target). Alternatively, you can use reverse inventory correspondence analysis which determines what targets were attempted by the child, and how each was realized in the child's pronunciations (Target -> Transcription).

These inventory analyses are designed to work at the level of context free "segments" rather than at the level of features, so the analysis proceeds ignoring diacritic information, and the printouts indicate the segments produced and their targets as displayed in Appendix 5 (no diacritics).

On the first page of Appendix 5 you'll find a Target -> Transcription analysis for DEMO1.LIP. Each segment listed on the left side of the page (each target) is associated with a row of corresponding segments as well as an indication for each of the number of times in the analyzed sample that the target was produced that way.

On the second page of Appendix 5 is a Transcription <- Target analysis printout, which indicates how often each element that the child produced came from the various targets s/he was attempting to produce in the file represented by DEMO1.LIP.

Although context free inventory correspondence analysis is designed to operate at a segmental level, it can function featurally to some extent, because LIPP offers many symbols that represent specific features. What it cannot do is to function context sensitively, that is to specify the occurrence of the various elements in initial position, in final position, after high vowels, etc.

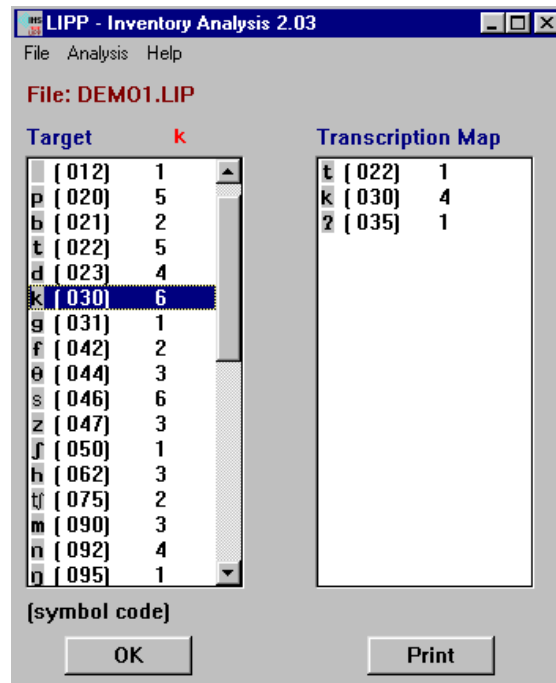


Figure 9.2 – Inventory Analysis Module (Last item in the Analysis Menu, Figure 9.1)

9.1.2 - Feature counts (context sensitive). **UL/LL** If you wish to count the number of occurrences of different features in a child's speech, or if you want to do context sensitive analysis of segment occurrences, LIPP offers a procedure that automatically counts the occurrence of phonetic features or segments. Kim provides an example of how this kind of analysis works (based on a file called KCsInvCt.LAL) or you can do this kind of analysis using an inventory you specify in a way we'll discuss below. If you only wish to do inventory feature counts on a child's speech, you don't really need the glosses or target forms to be in the transcript. Those Rows can be blank throughout.

9.1.3 - Feature correctness analysis:

If you want to do "Correctness" analysis, another LIPP analysis option, you also need the target forms. LIPP will automatically determine how often the child produces each of a number of predetermined elements (either features or segments, and if the user wishes to write the rules with context specification, that's possible) correctly and will specify the percentage of times each is correct. Kim provides a file to perform correctness analysis (KCorrect.lal). If you run this file to analyze a transcription (and the procedure for that is indicated below), you will note that each of a number of features and segments is specified in the results and the number of times they occurred

correctly in the child's speech is indicated as a proportion of all the times they occurred in target forms.

9.1.4 - Phonological Process analysis:

But if you really want to do a traditional process analysis, where featural specification is used generally, and where context sensitivity is specified, you are in business with LIPP. LIPP was designed to handle phonological process analysis naturally. In fact that was perhaps the primary purpose in the initial development of LIPP.

So the fourth analysis type offered by LIPP is "Process" analysis. Here, we utilize any of several rule sets offered by Kim, each designed to facilitate some important kind of analysis. The rule sets consist of phonological processes that LIPP will automatically search for in each transcription file you ask for. Among the processes available for search in the rule set called KCommon.LAL (Appendix 6) are cluster reduction, final consonant deletion, and a variety of additional processes representing common error types of childhood phonology. In addition, the rule set searches on a variety of less common error types, so that you will have the opportunity to evaluate odd characteristics of a particular child's system. Kim also provides Kadeqacy.LAL which is also a process analysis file.

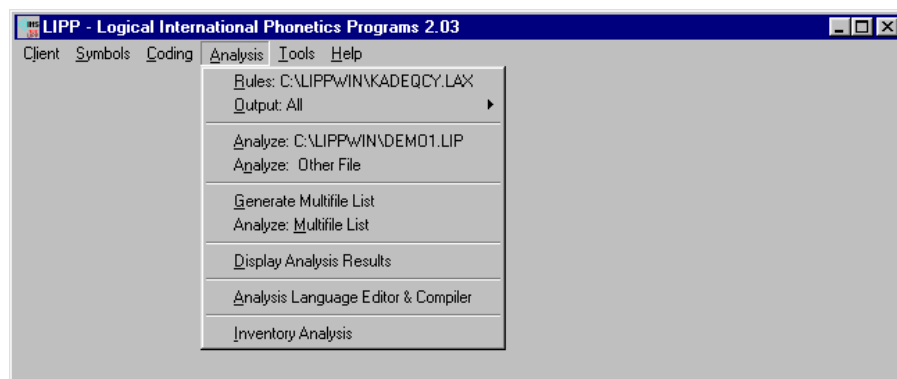


Figure 9.3 – LIPP analysis menu from Main LIPP module.

9.2 - Performing an analysis in LIPP: (UL/LL/SL)

It's actually quite easy to run analysis. Let's do one. Wherever you are in the menus or transcriptions, go back to the Main LIPP module and select "Analysis". The first option under Analysis will appear as the word "Rules" followed by the name of a rule file, or "None". The rule file we are now going to put there is KCommon.LAX. If for some reason you have a different rule file there, don't worry, we'll change it. Select "Rules" and hit <Enter>. You should then see a menu of all the available rule files, those that have a .LAX extension (the compiled ones). Select (using the "down arrow" key) the file KCommon.LAX. Now, still in the Analysis menu, select "Output". Now each time you hit <Enter> you will see an option appear for the kind of output you want (All, Occurrence, Summary, Report Only). Keep hitting <Enter> until "Summary" comes up, then stop.

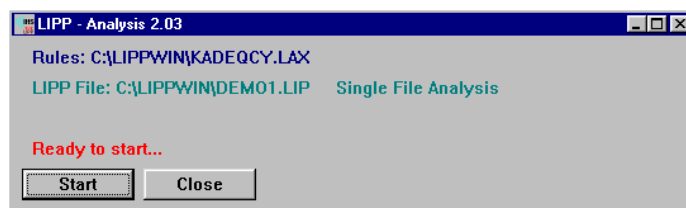


Figure 9.4 – LIPP analysis start dialog box. This dialog box will appear when either the first or second analysis option is selected. To start the analysis, press the [Start] button. After the analysis is completed, a [Display Results] button will appear. (see Figure 9.5)

Now you need to choose what file you wish to analyze. Under the first "analyze" option will be the name of the last transcription file that was examined or modified. To analyze that file with KCommon.Lal, you merely position the cursor there and hit <Enter>. The analysis will begin, and it will take a while because this is a big analysis file. Similarly you can choose the second "analyze" option ("other file") and a new menu will appear listing transcription files (files with .LIP extensions) that you might wish to analyze. Select DEMO1.LIP, and your analysis will be performed, with the results displayed on the screen, as the analysis proceeds.

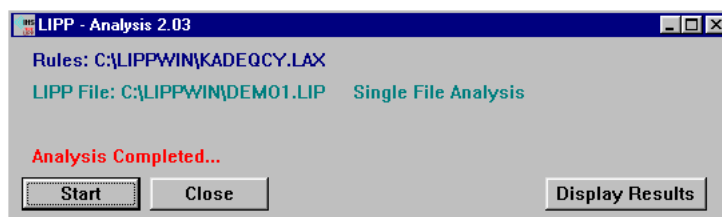


Figure 9.5 – Analysis dialog box showing [Display Results] button after completing an analysis.

9.2.1 - Displaying results of an analysis to the screen:

If you conducted your analysis from the first or second "analyze" options in the Main menu under "Analysis", then at the end of the running of the analysis, the display analysis button will be activated allowing you to select it and view the analysis results.

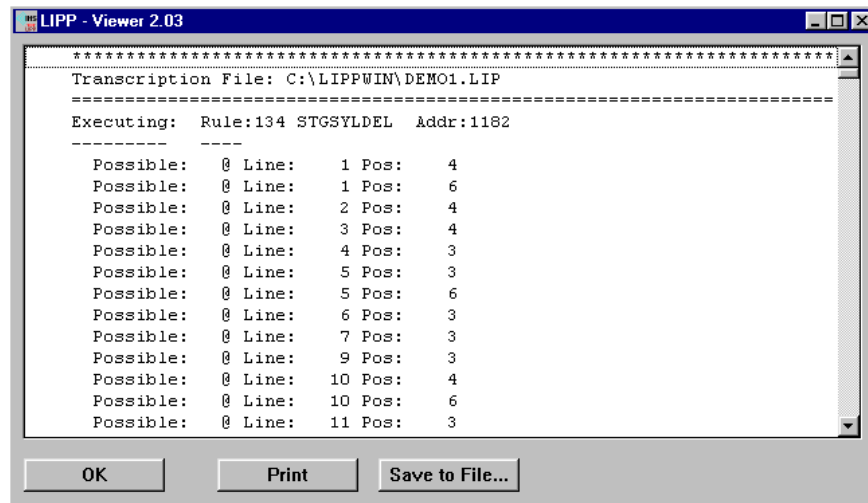


Figure 9.6 – Analysis results viewer module. [Print] and [Save to File...] buttons may be used to print and save the currently displayed results to a file for further analysis. Scroll bars along the side may be used to view all results.

9.2.2 - Printing results of analysis:

After performing an analysis and selecting the [Display Results] button, you can print the displayed results by pressing the [Print] button.

9.2.3 - Saving results files: (UL/LL)

A results filename is automatically assigned to a results file based upon the filename of the analysis file [they just have different extensions] each time you run an analysis. Consequently, if you run another analysis with KCommon.LAL, on a different transcription file, then the KCommon.RST file for DEMO1.LIP would be written over. If you wish to save a particular results file permanently, you can print it out, or you can save it to another file for permanent storage. LIPP includes another method of saving a results file under a special name.

9.3.4 - Other analysis examples:

The other kinds of analysis in LIPP are equally easy to run. For Inventory Feature counts, you simply select the file KCfInvCt.LAX (for context free segmental and featural analysis of consonants) or KCsInvCt.LAX (for context sensitive segmental and featural analysis of consonants) from the Analysis menu (you will have to Quit the Editor menu to get back to the Main menu if you went into the Compiler & Editor for the printing or displaying of results). For Correctness analysis you select KCorrect.LAX.

Segmental Inventory Correspondence analysis is from a special location: select the last item, "Inventory Analysis", from the Analysis menu (the one where you presumably are now, the one you reach via the Main menu, rather than the one you reach via the Editor menu). Let's do one of these also. When you select "Inventory Analysis" you will see a new menu (the Inventory Analysis menu) that will ask you to specify the name of the "Transcription File" you wish to analyze, and the name

of the "Output Data File" you will create. Select the first option ("Transcription File Name") and type in DEMO1.LIP <Enter>. (You may also press <Enter> to get the list of available .LIP files). Then select the second option ("Perform Inventory Analysis"), and the analysis will be run. When it finishes, the screen will tell you so. Then select the option "View Inventory Results (Trans -> Target)".

At this point, a new screen will appear with the first Line of the inventory analysis results. By using the "right arrow" key, you can advance the results file Line by Line, to look at a record of what sounds the child produced and what target segments they were based on ("originated from" in Target Forms). Page forward one page with the "right arrow" key to see the results on the segment [p]. It was produced correctly just once, it was produced as an adjunction once (i.e., it originated from a Nil or blank space on a Target Form Row), and it was produced in substitution for an [m] once. There were three [p]'s produced in the sample by the child.

All the segmental elements in the pronunciations transcribed in the file (DEMO1.LIP) are accounted for in the file you should be viewing page by page using the "right arrow" to advance. The first page shows the Nils, produced by the child, which means blank spaces in the child's transcript, where s/he deleted a segment, and all the segmental targets upon which each of these Nils or deletions were based. On all subsequent pages, the child's inventory segments are shown on the top line and the targets upon which they were based will be listed (on subsequent rows below the first one on each page). The numerical values or "counts" at the right of each symbol will indicate the number of times each element was produced correctly or produced in substitution for another element. The very first row (the one lit up in red when you first view each page) represents a child inventory segment, and its numerical value tells the number of times that segment was used correctly. The remaining rows on the screen show what other segments were targets when the child produced the specified inventory segment, and how many times.

To leave "View Inventory Results", hit <Esc>.

When the Inventory Analysis menu appears, you select "View Inventory Results (Target -> Trans)", and then you get a display that shows Target Forms on the first rows and the child substitutions on the lower rows. You can then page through the outcome results the same way as indicated above.

To print results of an Inventory analysis or Reverse Inventory analysis, simply select the "Print Inventory Results" option in the Inventory Analysis menu after you have gone through the analysis steps. Both the Target -> Trans and the Trans -> Target analyses will be printed.

To leave the Inventory Analysis menu, hit <Esc>.

9.4 Reliability Analysis:

A particularly interesting kind of analysis that can be performed in LIPP is an analysis of transcription reliability. For example, suppose that two transcribers are working on the same tape.

They work independently and when finished, they look at the two transcripts and notice some discrepancies. How do they analyze them? How do they quantify the degree of reliability. Similarly, suppose one transcriber does the same tape twice, with a several week delay in between the two transcripts. Then that person may wish to analyze his/her own intratranscriber reliability. How consistent were the two transcripts?

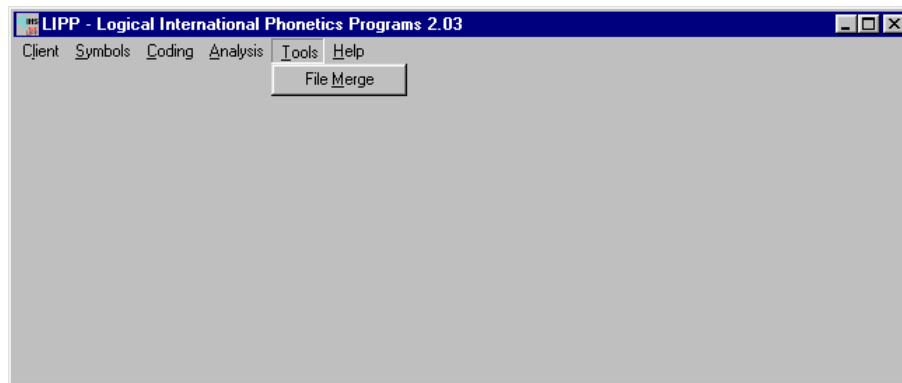


Figure 9.7 – Tools menu from LIPP main module.

LIPP assists transcribers in this kind of reliability analysis. The first step is to create a combined or "merged" file, using the "File Merge" option under "Tools" in the MAIN menu. Following the instructions in the succeeding menus (see Section on "File merge") you create a new file that puts transcript 1 (the one to be treated arbitrarily as a standard) and transcript 2 (the one to which transcript 1 will be compared) together. Which is to be the standard, and which the comparator, is the decision of the user. The Transcription Rows from transcript 1 appear on the Target Form Row of a new "output" file, and the Transcription Rows of transcript 2, appear on the Transcription Rows of the new file.

The process of merging the two files is very easy if the two transcripts are very similar, but it is slower if the two transcripts are very different. Merging has to be done thoroughly and accurately (see Section "Merging files for reliability analysis") or the reliability analysis will not be correct.

Now to conduct the reliability analysis, you can run an analysis using a special file (KReliab.LAX) prepared by Kim. The output will show you how often the two transcripts agreed or disagreed on a number of features preselected by Kim. If you want to have additional features considered, you can write additional rules to make that possible (see below.) The results of a reliability analysis, plus the reliability analysis rules from KReliab.LAL are displayed in Appendix 8 based on a real merged file from two transcribers, RelK&L.LIP, found in Appendix 9.

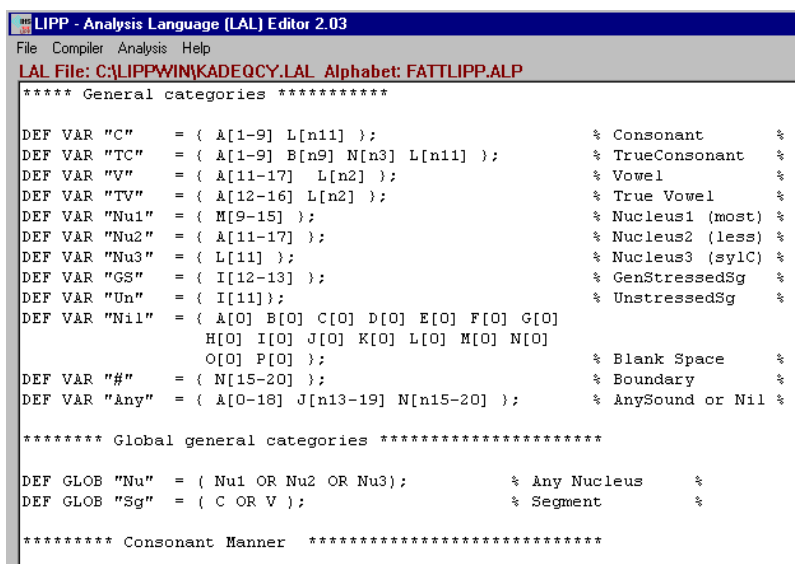
10 - How LIPP analysis works (UL)

Process analysis in LIPP is based upon the comparison of Transcription Rows with Target Form Rows. That's why when you are typing in transcriptions you must line up the elements and spaces appropriately. If the target word is "stop" (with four segments) and the child says [top], (with three) then the user of LIPP must decide where to place the space on the Transcription Row. If the space goes under the [s], LIPP rules (as currently written, such as those in KCommon.LAL) will interpret the change as an [s] deletion, but if the space is under the [t], the rules will read the change as a deletion of [t] and a stopping of [s].

Consequently, whenever you are transcribing with the intent of having a phonological analysis done, it is necessary to line up Target Forms and Transcriptions in a way that is consistent with the interpretation you give to the changes. In some cases, actually, rules can be written so that you have more than one option in terms of how things can be lined up, but those cases are special ones. Usually you should make your alignment decisions as you work.

Correctness analysis and Inventory analysis also operate in a way that depends on proper alignment of symbols, so it is important to learn to line things up from the very beginning.

From here until the section on LIPP Symbolology Development, we're going to discuss the way LIPP does analysis. If you want to skip this section, it's OK, but this is the place to learn how to write your own rules.



```
***** General categories *****
DEF VAR "C"      = { A[1-9] L[n11] };           % Consonant      %
DEF VAR "TC"     = { A[1-9] B[n9] N[n3] L[n11] }; % TrueConsonant %
DEF VAR "V"      = { A[11-17] L[n2] };          % Vowel            %
DEF VAR "TV"     = { A[12-16] L[n2] };          % True Vowel      %
DEF VAR "Nu1"    = { M[9-15] };                 % Nucleus1 (most) %
DEF VAR "Nu2"    = { A[11-17] };                 % Nucleus2 (less) %
DEF VAR "Nu3"    = { L[11] };                   % Nucleus3 (sylC) %
DEF VAR "GS"     = { I[12-13] };                % GenStressedSg   %
DEF VAR "Un"     = { I[11] };                   % UnstressedSg    %
DEF VAR "Nil"    = { A[0] B[0] C[0] D[0] E[0] F[0] G[0]
                  H[0] I[0] J[0] K[0] L[0] M[0] N[0]
                  O[0] P[0] };                 % Blank Space     %
DEF VAR "##"     = { N[15-20] };                % Boundary        %
DEF VAR "Any"    = { A[0-18] J[n13-19] N[n15-20] }; % AnySound or Nil %

***** Global general categories *****

DEF GLOB "Nu"    = { Nu1 OR Nu2 OR Nu3 };        % Any Nucleus     %
DEF GLOB "Sg"    = { C OR V };                  % Segment         %

***** Consonant Manner *****
```

Figure 10.1 – LIPP Analysis Language (LAL) Compiler screen. The LAL compiler is used to modify and write new rules.

10.1 - Definition of phonetic terms or variables in LIPP:

How does LIPP know what it means to be a "plosive" or a "consonant", and how does it do its counting? You may have already guessed that the LIPP Analysis Language (LAL) understands such terms by reference to the 16 dimension coding. Rafael wrote LAL to make it possible for a user who knows nothing about computers to write rules to do any of the kinds of LIPP analysis listed above, by simply following a few simple procedures that yield rules for LAL to interpret and implement.

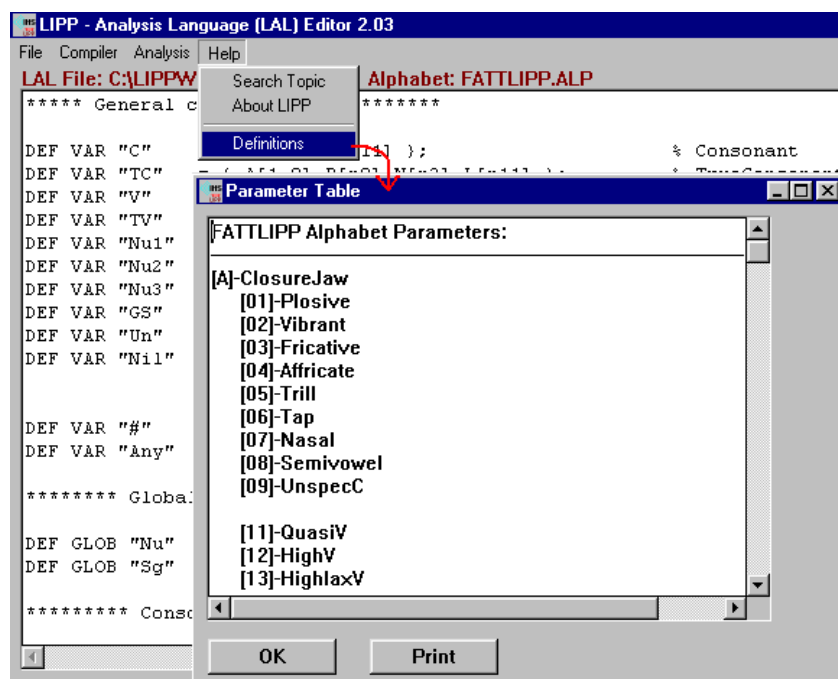


Figure 10.2 – Symbol Definitions table. You may access the symbols definitions by pressing the [Definitions] option in the help menu of the LAL compiler.

Actually there are two steps in the process. First you need to define your terms. For example, you may wish, in rules that you write, to make reference to the class of segments that are called "consonants". You can do this for LAL by writing a "variable definition" for the term "consonant", let's abbreviate it to "con". The definition might look like this:

```
DEF VAR "con" = { A[1-9] }; % consonant %
```

This statement says we have defined a variable "con" to mean any element that has a value from 1 to 9 on the A dimension in the 16 dimension code. The comment "consonant" is enclosed in % signs, indicating to LAL that it is just a comment, and thus does not have to abide by LAL syntactic rules. Take a look in Appendix 2 at the code chart, and you'll see that the A dimension from values 1 to 9 indicates a series of consonantal features (plosive, vibrant, fricative, affricate, etc.). In any of Kim's alphabets, an element that is a consonant will indeed have a value from 1 to 9 on the A dimension.

Now let's define the notion "nasal consonant", abbreviated "nas". It's even easier.

```
DEF VAR "nas" = { A[7] };    % nasal consonant %
```

LAL lets you write these definitions in a general way, that includes "boolean" functions, like "not", that make it even easier. Suppose you wish to define the notion "consonant, but not syllabic consonant", let's abbreviate that to "CnotSyl".

```
DEF VAR "CnotSyl" = { A[1-9] L[n11] };
```

This variable will be understood by LAL as an element that has a value from 1 to 9 on A, and a value on L that is not 11 (the value that indicates syllabicity of consonants).

You can define many different terms in LAL, and in fact, you can even define terms in terms of terms. (Kim loves that kind of talk). A "global" term, for example, is defined in terms of previously defined terms. For example, if you wish to define the notion "OralConsonant" (this time we're not abbreviating, because we don't have to if we don't want to, and LAL will understand anyway), you could use the following definition statement:

```
DEF GLOB "OralConsonant" = ( con AND NOT nas );
```

LAL understands this to mean that you want to define an oral consonant as an element that is a "con", as previously defined, and is not also a "nas", as also previously defined.

See Appendix 10 for the formal syntax requirements of variable and global definitions in LAL.

If you look into Appendix 6 you will find variable definitions for KCommon.LAL, followed by global definitions and finally phonological process definitions.

When you write definitions in LAL you have to make sure that any Globals you define utilize only previously defined terms, i.e., the terms used to define a Global must have been defined on the list before the global definition appears. Otherwise LAL doesn't know what you're talking about.

Here are a few additional useful definitions:

```
DEF VAR "vow" = { A[11-17] };    % vowel    %  
DEF VAR "fri" = { A[3] };        % fricative %  
DEF VAR "sto" = { A[1] };        % stop     %
```

10.2 - Writing rules in LAL:

Now to write a rule using previously defined terms is easy. Using the terms that have already appeared on our list above, let's write a couple of useful rules.

```
DEF RULE "Stopping"      = (( fri ) -> ( sto ));  
DEF RULE "Nasalization" = (( OralConsonant) -> ( nas ));
```

These rules indicate that "stopping" is to be interpreted as a substitution of a stop for a fricative (in any position), and that "nasalization" is to be understood as a change of an oral consonant to a nasal one. The rules use a syntax that allows any previously defined terms to be referred to. It is important to remember to put a semicolon at the end of each rule, and to enclose each rule in parentheses. Furthermore, you need to enclose the statements on each side of the arrow in parentheses.

Sometimes we actually write rules that are not intended for direct execution, but are rather going to be used as parts of other rules. If we do want a rule to be executed, the statement we use is DEF EXEC RULE, not just DEF RULE.

Setting up a simple analysis. Let's now go through the steps of doing an analysis together. First enter the MAIN menu, and select "Analysis". Then scroll down to "Compiler & Editor" and select it. A new screen will appear, Rafael's editor screen menu. Using the "left arrow" key, go across to "Edit" and select it by hitting <Enter>. The top line of the screen will turn to red when you are in the edit mode. Your cursor will now appear on a blank page (except for the menu information on the top line), and you are ready to write LIPP definitions and rules (as long as the top line is red -- if it isn't then you need to select Edit and hit <Enter>).

Enter the following definitions and rules, the ones that we described above. If you feel more comfortable using your own editor or wordprocessing system to write rules, you may do this. Just remember to use a NONDOCUMENT mode for all rule sets and give the file you create a .LAL extension. When you're done, you go back into LIPP and compile the rules you wrote without using LAL's editor.)

This is how your page should look when you're done typing:

*** My first LIPP rules *****

```
DEF VAR "con" = { A[1-9] }; % consonant %  
DEF VAR "nas" = { A[7] }; % nasal consonant %  
DEF VAR "CnotSyl" = { A[1-9] L[n11] };  
DEF VAR "vow" = { A[11-17] }; % vowel %  
DEF VAR "fri" = { A[3] }; % fricative %  
DEF VAR "sto" = { A[1] }; % stop %
```

```

DEF GLOB "OralConsonant" = ( con AND NOT nas );

DEF EXEC RULE "Stopping"      = (( fri ) -> ( sto ));
DEF EXEC RULE "Nasalization" = (( OralConsonant) -> ( nas ));

END RULES;

*****

```

The END RULES statement with the semicolon is necessary, so that the LAL compiler knows where to stop its work. When you have finished typing the definitions and rules (you don't have to put in the asterisks or the title line ("My first LIPP rules") if you don't want to), but sometimes the asterisks are nice to set off different sections of your rule set. Remember that anything you write that is not to be interpreted as a definition or a rule, must be enclosed in % signs, or the line on which it occurs must begin with an asterisk. These are the symbols that LAL's compiler understands as meaning "ignore this part".

Now let's save the file you created. Select "Files" and then "Save File". When you are queried for a file name, call it something that isn't already likely to be in the .LAL directory. How about calling your file MyFirst.LAL. Now select "Compile" and the compiler will go to work. If you typed things in just the way they are above, the file will compile without errors and you'll be ready for analysis.

If you made any important mistakes typing in the rule set, the LAL compiler will probably find them and tell you something about what they were. For example, if you failed to put in the right number of parentheses in a rule, LAL will tell you that your parentheses are "unmatched". (You must have the same number of left and right parentheses). Most other error types will also be specified. The last line of the compilation statement will indicate the error, and the last line of what was being compiled when the error was found by the compiler will be on the screen. Look at that last line of your definition/rule set because that's where you need to go to when you go back into the editor to correct your mistake.

If you did make a mistake, hit <Enter> to get out of the compiler, and use the left arrow key to get back to "Edit". Select "Edit", the top line will turn red to indicate you are now editing, and scroll down to the spot where the error was and correct it. Now hit <Esc>, the top line will turn black to indicate you are no longer in the edit mode, go to "Files" and then "Save Program File" using the same filename as before (it will occur as a default value if you just hit <Enter>), and then go to "Compile" and try again.

Keep following this procedure until the compiler tells you it completed its job. That will mean you're ready for analysis.

There are two ways to do your analysis. One is to select Analysis directly from the Editor menu. The analysis will be conducted using the file that is currently loaded, assuming that it has been compiled. This is the analysis method that is the most convenient if you are debugging your analysis files, because you don't have to go out of the Editor. Another reason for using the Compiler & Editor to run Analysis while you are debugging is that it offers you the option of running the analysis on selected Lines of transcript only. Suppose the error you are trying to correct in your rule set is best tested by lines 10-12 of a particular .LIP file -- the whole file might take several minutes, but the analysis of just lines 10-12 could be extremely fast. So you select "Options" from the Compiler & Editor menu, and then select "Start line:" by hitting <Enter>, then entering the value "10" plus <Enter>. Similarly you set the "End line:" to 12. Then select Analysis (still in the Compiler & Editor, because the Line limitation only works from there), and the test of your debugging question will be conducted very quickly.

More commonly, however, when you are running analysis for the purpose of obtaining data on a real client, you will not do it from the Compiler & Editor menu, but from the Main menu. Use the "right arrow" key to go to "Quit", in order to get out of the Compiler & Editor and back to the MAIN menu. Select "Analysis". Go to the first line, "Rules:" and select it. You will see the list of available rule files. They will have extensions with .LAX, meaning they are compiled LAL files. Select your file, presumably the one you called MyFirst.LAL, and will now appear as the compiled file MyFirst.LAX.

Now go to the second "Analyze" option on the same Analysis menu and select that. You will see the list of available transcribed files. Select DEMO1.LIP. Now the analysis will begin. On the screen you will see the whole thing scroll out in front of you and what will be there will indicate that "stopping" occurred five times (and there were 16 possible occurrences, since there were 16 fricatives in the Target Forms), while "nasalization" occurred just once in 57 possible cases (where oral consonants occurred in the Target Forms). Your screen printout should look like the one in Appendix 11.

If you wish to obtain a hardcopy printout of the results of your analysis, go back to the Analysis menu and select "Print Analysis Results". Also you can print from the "Compiler & Editor". When the Editor menu comes up, select Files, and then "Print Results". Select the results file appropriate for the filename you created (perhaps MyFirst.RST).

If you view the whole results file as in Appendix 11, notice that LIPP records the outcome of each rule (EXEC RULE, not just RULE) in the order that it was entered, and looks Line by Line in the transcript for instances of the application of the rule. If it finds them, it so indicates with a Line number and Position (column of transcript) number to show you where things happened. For example, "stopping", as defined above, occurred in the DEMO1.LIP file in the second Line, second Position, and at the 13th Line, second Position, etc. You can look in the transcript of DEMO1.LIP (Appendix 4) to verify that fact. The last lines of the results file provide the summary of rule applications and an indication of the number of times the rule could have applied (the number of times that the structural description, or left side of the rule, was found) as well as the number of

times it actually did apply. These values are expressed as a ratio, and the percentage to the right of the ratio can be interpreted as a measure of how often the child uses the rule in question.

(c) Other analysis features. If you want your analysis to output all the Line by Line data on occurrences or applications of the rules, go to the Analysis option of the Main Menu, select "Output" and hit <Enter> until "Occurrence" appears. Then when you analyze, LIPP will provide full a record of each Occurrence, Line by Line. But if you just want the Summary, you can choose that option in the Analysis menu under "Output:" by hitting <Enter> until the Summary option appears. Then your screen display, or your printout, will only show the summary rather than the Line by Line analysis for each rule. A third option is to print the SUMMARY plus the OCCURRENCE, plus the Line by Line data on the places that the rule did not apply even though the structural description (the left side of the rule) was satisfied. To get this maximum display of the analysis, select ALL under "Output:". You will sometimes need to use ALL output format when you are debugging, because it provides the most convenient way to find the cause of errors in the analysis file.

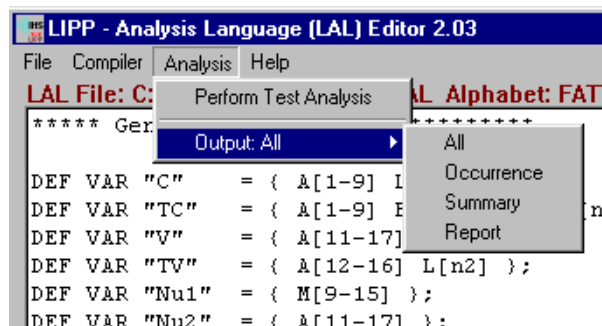


Figure 10.3 – Selection of output options from the Analysis Menu in the LAL compiler module for analysis results. This feature is also available from the Main Module, also in the Analysis Menu.

The simplest of all the output formats is the REPORT ONLY, also selectable from the "Output" option. Not all analysis files have reports, so in some cases, selection of that option would be inappropriate. But for analysis files that do have reports, generated to present data in some format you may find handy, the report may be all you care to see. (See more about reports under "Report Generation", below.)

You can look at a results file by entering the Compiler & Editor from the Analysis menu and going to "Files". There is a length limitation on the "Load Program File" function -- only the first 750 lines of a file (whether it's a results file or a program file) can be loaded -- however, if your results file is 750 lines or less, the following is a convenient approach. Select "Load Program File", hit <Enter> twice and respond to the path query with *.RST. The new menu that appears will be the list of results files (rather than LAL or "program" files), so you scroll to the name of the rule set you used (for example, MyFirst) but this time with the .RST extension, and select it. Then use the "right arrow" to select Edit and hit <Enter>. Now you can view the file in the editor, scrolling up and down.

Anytime you run an analysis, a results file is created with a name that corresponds to the main part of the name of the rule set you used plus an .RST extension. When you are in the editor, you can scroll through the results file on the screen to make sure all the rules you wrote are being applied the way you meant for them to be. But if your RST file is too long (750 lines is the maximum), the editor won't be able to read the whole thing for display on the screen, and you'll need to go out of LIPP and look at the RST file in a word processor (WORDSTAR or WORDPERFECT for example) in order to see it all. You can, however, page through the whole long file using <Page Down> for one page at a time or <Enter> for two pages at a time (although you will not be able to scroll backwards) by selecting the "Display Results" option under "Files" in the Editor menu, and you can always print the entire file to paper using the Print Results option in the same location. The same print option is available under "Analysis" in the Main menu if you select "Print Analysis Results", and the display option is available under "Display Analysis Results".

10.4 - A few quick tips on advanced LIPP rules:

The best way to learn some advanced LIPP rule writing is to examine the rules and definitions provided in some of the Appendices. You'll notice there that you have the option of writing rules that combine rules. For example a combination rule like this one is acceptable:

```
DEF RULE "ConsonantError" = ( Stopping OR Nasalization );
```

In addition, one can use the boundary symbol or other previously defined terms to write context sensitive rules such as this one:

```
DEF RULE "InitPrevStopng" = (( # fri V ) -> ( # sto V ));
```

This rule says that a fricative following a word boundary (that is, in initial position), and preceding a vowel, is changed to a stop in the same phonetic environment.

When you write definitions of terms, you can refer to values of segments on particular dimensions of the 16 dimension code, as indicated above, or if you wish you can write definitions based on the ordinal position of the segment in the alphabet you are working with. See Appendix 3 for the TITELIPP alphabet listing, and note that each symbol (main symbols, diacritics, globals, boundaries, numerals) has an ordinal position. [p] is number 20, [b] is number 21, # is number 255. We can define in terms of those ordinal values. For example, the following is a well-formed LIPP definition:

```
DEF VAR "Bee" = { CHR[21] };
```

This definition defines the character [b] (calling it "Bee") for subsequent rule reference. Similarly,

```
DEF VAR "AspPee" = { CHR[20] DIA[187] };
```

defines the aspirated [p] as character number 20 and diacritic number 187, the superscript "h".

Definitions such as these are very convenient if you wish to do special segmentally specific analysis.

10.5 - Assimilations, metatheses, coalescences:

The writing of rules to implement notions such as coalescence, metathesis, or assimilation requires context sensitivity in rule structures. All these types of rules are possible in LIPP. Examples are provided in the file AsmRule.LAL, available in UPPER LIPP systems. To see the rules work, try them out on Demo3.LIP, a file that includes examples of assimilation, coalescence and metathesis.

10.6 - Exact length rules and recursive definitions:

Assimilation rules as written by Kim, utilize Exact Length rules (look at the rules in AsmRule.LAL by entering the Compiler & Editor and loading that rule file) and Sequence definitions (see "sequences" among the global definitions in the AsmRule.LAL file) that include recursive definitions. Exact length rules are designed to make sure that as LIPP looks for applications of a rule, it looks at precisely the same domain in the target form and the transcription (i.e., the columns evaluated for the structural change are exactly the same columns indicated in the structural description). This is necessary in some cases where definitions of sequences of elements (for example "consonant groups" see AsmRule.LAL) can be of varying lengths. Sequence definitions, in order to be stated very generally, require a recursive definition in which the element being defined is named in its own definition. For anyone who plans to write recursive definitions, it is important to note that there are two key syntactic constraints: the name of the element being defined must occur last in any intraparenthetic sequence where it occurs in the definition, and the last element mentioned in the whole definition must not be the element being defined. Thus, based on these definitions,

```
DEF RULE "ConsGroup" = (( Con OR ConsGroup ) OR Con );
```

is a well-formed recursive rule defining a sequence of consonants of indeterminate length, but

```
DEF RULE "ConsGroup" = (( ConsGroup OR Con ) OR Con );
```

is not well-formed, nor is

```
DEF RULE "ConsGroup" = (( Con OR ConsGroup ) OR ConsGroup );
```

Errors in the definitions of sequences can cause the analysis you run to hang up in loops, so you must be careful with recursive rule syntax. Study the "sequence" and "expanded sequence" definitions in AsmRule.LAL to get a clearer idea as to how such recursive definitions are written.

10.7 - Calculation metrics:

LIPP allows one to calculate a metric of phonological development or reliability based on the data in any analysis file. For example, consider an analysis that determines that initial stopping (InitStop) occurred 10 times out of 100 possible occasions in a particular sample, that final stopping

(FinStop) occurred only once in 100 opportunities and medial stopping (MedStop) occurred 5 times in 100 opportunities. One could write a calculation rule that determines the overall occurrence of stopping in initial, medial and final positions. The purpose of such a calculation might be to get a general idea about stopping, but to do it in a way that allows the user to implement a theoretical belief that initial stopping is much more damaging to intelligibility than medial or final stopping. LIPP allows the user to weight the initial stopping more heavily than the medial and final stopping. In fact the user can choose whatever weights seem appropriate. Suppose the user believes initial stopping is twice as bad as stopping in other positions. The following "display calculation" rule would be appropriate:

DISP CALC "Genlstopping" = ((2 * InitStopP) + (MedStopP) + (FinStopP) / 4);

The rule instructs LIPP to display the value of two times the initial stopping percentage (that's what the final P stands for and in this case it's value would be 10%), plus the medial stopping percentage (in this case 5%), plus the final stopping percentage (1%), all divided by four. In this way initial stopping is given twice the importance of the other two values. The user of LAL can write calculation rules with great flexibility because the system allows specification in terms of the four basic arithmetic functions, and allows the user to refer to the outcome of application of rules in terms of their percentage (indicated by a P at the end of the rule name), their occurrence or number of occasions where the rule applied (indicated by an O at the end), or their count (indicated by a C) which is to say the number of occasions where the structural description of the rule was met in the speech sample. The percentage of application is of course the ratio of O to C.

The primary use that we see for calculation rules is that they allow us to implement metrics of phonological development or reliability measures that are adjusted to whatever assumptions the UPPER LIPP user is interested in making. For example, one phonological process may be much more problematical to the child than another -- the calculation rules allow the user to assign greater weights to the more problematical rules. By writing a sequence of calculations and then combining them in calculations based on the outcomes of previous calculations, the user can create a general metric of phonological development, focussing on whatever processes s/he chooses and weighting them to whatever extent s/he deems appropriate. Both KCommon.LAL (see Appendix 6) and KAdeqacy.LAL include calculation rules.

For reliability analysis, one can create calculation rules that are based on the degree of discrepancy between two transcriptions. An analysis of reliability begins with a set of rules that compare two transcripts of the same set of utterances (such a comparison requires file merging, which is discussed below under "File Merge"), determining how the two differ. Calculations of reliability can then be focussed on the outcomes of the rule applications and can determine how similar or dissimilar the two transcripts were based on a metric that the user can design, again with the possibility of weighting some transcription discrepancies (i.e., rule applications) more heavily than others.

For many years reliability measures reported in journal articles about child phonology have tended to indicate an overall transcription reliability measure. But usually, it is not clear what the measure includes. LIPP makes it possible easily to specify exactly what reliability measures one uses. Further it allows us to create much richer reliability measures that focus on various levels of reliability and allow consideration of different kinds of discrepancies, while not preventing the user from having an overall measure of reliability. In fact, LIPP's measures can be especially helpful because LIPP offers the user the opportunity to judge the relative importance of different kinds of discrepancies, and to incorporate that judgment in the automatic calculations. Since the data from calculation analyses are available in disk copy or hard copy, and the method of calculation can be printed out, the user has a firm record of how reliability is calculated on each occasion, and can use precisely the same assumptions on other occasions if desired -- all automatically. If the assumptions change, all that is required is rewriting the calculation rules to implement the new assumptions. KReliab.LAL (Appendix 8) provides an example of one implementation of reliability analysis with calculation metrics to weight various kinds of discrepancies.

Another application of calculation rules is in the evaluation of certain ordering constraints on rules. For example, in a "bleeding order", one rule applies, and another rule that would apply had the other rule not been applied first, is left out. Consider the following scenario: a child says "dog" as [da] and "doggie" as [gagi]. One possible analysis of this arrangement is that there is a final consonant deletion rule (FCDel) that "bleeds" the velar assimilation rule, so that the assimilation occurs when the target [g] is medial, but not when it is final. We might say that FCDel precedes velar assimilation, eliminating the environment of its application in words where the target velar is final. Velar assimilation then occurs in some cases but not others. We specify two rules of velar assimilation, VelAssim1, which accounts for final position target velars, and VelAssim2, which accounts for nonfinal position target velars. To evaluate the bleeding order constraint, a user might write the following rules:

DISP CALC "BleedingDel" = (VelAssim2P / VelAssim1P);

This rule asks for the ratio of assimilations of nonfinal to final velars. If the ratio is large, it suggests that there is a bleeding order relationship. If the ratio is near to or less than one, it suggests no such relationship. The user might decide to adopt as a criterion that the ratio must exceed two in order for the ordering criterion to be met.

10.8 - Phonetic avoidance analysis in LIPP:

It is widely reported that children sometimes have lexicons that systematically avoid words that have some phonetic property. For example, a child might refuse to pronounce words that begin with a consonant, or begin with a fricative. Lexical or phonetic avoidances can be analyzed quite simply in LIPP. We invoke the sequence definitions of a Nil Group or "NilGp", meaning a sequence of blank spaces, and "AnyGp" meaning and sequence of segments. Suppose we formulate a lexical avoidance rule for words with initial consonants as follows:

DEF RULE "InitCAvoid" = ((# C AnyGp #) -> (# NilGp #));

The rule says that a consonant following a word boundary becomes a whole nil word, meaning that the word is not pronounced at all. Similarly

```
DEF RULE "InitSAvoid" = (( # S AnyGp # ) -> ( # NilGp # ));
```

says that if a word begins with a spirant, it is not pronounced. A LIPP analysis using lexical avoidance rules of a transcript based on a particular assessment instrument (with target forms to be requested of the child already entered), where the child pronounces some words but says nothing in response to others, can indicate the number and proportion of cases where a child avoids any specified phonetic property.

10.8.1 - Report Generation:

A brand new feature of LIPP version 1.40, is the ability to create reports at the end of an analysis, based on calculations from the various rules applied, from demographic information stored in the file, and displayed in a format that the user can choose. Appendix 12 includes a report on a babbling file called Baby.LIP, based on the analysis file Canon.LAL (all the relevant documents are included in the Appendix). Canon.LAL is merely intended to show how reports are programmed in LAL.

The rules are quite simple. A "Display" statement must occur after all Definitions of Rules, Variables, Globals, and Calculations that it references. In the simplest form, a Display statement consists of the word "Display" followed by whatever words one wants the report to include on a particular line, typed in quotation marks, and followed by a semicolon as a line delimiter. Thus

```
Display " This is my babbling report ";
```

would create a line of text at the beginning of the Report saying
"This is my babbling report", with two spaces at the beginning. The command

```
Display " ";
```

would generate an empty line on the report.

Display statements can directly reference certain demographic information that may have been entered for the transcription file in question. One enters such information following a comma after the closing quotation marks, in parentheses, after the characters DEMO_ (the underline is important). Thus

```
Display " Name of child: ", (DEMO_Name);
```

is a well-formed Display statement that will generate a line in the report with the words " Name of Child:" followed by the name that is found in the transcription file under the demographics slot for

"Name". Note in Appendix 12 that the Report printed out shows the name that is in the file Baby.LIP. The available terms that can be referenced in demographics from Display statements are Name, ID for (Local ID number), BDate (for birthdate), SDate (for session date), Age (for age of the child in days), Tran (for transcriber), Disa (for disability), Lang (for language), Gest (for gestation), Comm (for session comment), Alph (for alphabet), File, and Sex.

Age and Gest can be used in arithmetic calculations. For example,

Display " Age of child: ", (DEMO_Age/365);

will convert the age of the child from days to years and print the result in the report (see Appendix 12). Multiplication, division, addition and subtraction are available within Display statements according to the same conventions provided for calculation rules.

Display statements can also make reference to any (Execution) Rule outcome or any Calculation outcome. Thus, referring back to the rule we wrote for Stopping,

Display " Frequency of occurrence of Stopping : ", (StoppingO);

would print in the Report the "occurrence" value (that's what the O stands for) of Stopping in the analyzed file. Display rules can similarly access "C" or "count" values (the number of times the structural description of a rule was met, for example, StoppingC) and "P" or "percent" values (the ratio of occurrence to count, for example, StoppingP).

If you take a look at the calculation rule in Canon.LAL for "CanBBRatio" or "Canonical babbling ratio", you'll see that it represents the number of "canonical" syllables in the baby's vocalizations, divided by the number of utterances in the sample. The statement

Display " Canonical Babbling Ratio ", (CanBBRatio);

will generate a report line including the indicated text, followed by the numerical value of the ratio. For real numbers that you do not wish to display to several significant digits, you may wish to truncate values, or round them off to the nearest unit digit. If so you can create a statement such as

Display " Ratio ", Trunc(CanBBRatio);

or

Display " Ratio ", Round(CanBBRatio);

Display rules can be written in such a way as to combine the calculation capabilities with regard to rule outcomes and demographic information. Thus

Display " Complex Score", (DEMO_Age/365 + (CanBBRatio * 10));

is a well-formed Display statement.

Report generation also includes "If/Then" statements. For example,

```
IF (( StoppingP ) > ( 10 )) THEN  
Display " Stopping :", Round(StoppingP);
```

would create a line in the report indicating the rounded value of the percent of stopping, if and only if that value exceeds 10. If/Then statements can invoke comparatives >, <, =, or sequences such as >= (which is read as greater than or equal to).

In the course of preparing a report, one may need to compare certain demographic values with outcomes of rule application in order to create a metric that one may wish to have printed in a report. This can get complicated, so LIPP version 1.40 offers the ability to make calculations that are stored in one of ten buffers (Buffer0, Buffer1..., Buffer9), the contents of which can be treated arithmetically. Consider the sequence,

```
Calculate (Buffer1: = StoppingO);  
DEF Calc "Age" = (DEMO_Age/365);  
IF ( ( Age ) < ( 4 ) ) THEN Calculate (Buffer1: = Buffer1 - .2 ); Display " Stopping score adjusted  
for age ", Round(Buffer1);
```

which puts the "occurrence" value of the "stopping" rule in Buffer1; then if the child's age is less than 4 years, the sequence of commands adjusts that value by subtracting 0.2 from it, and finally prints in the report the result of that calculation, labelled "Stopping score adjusted for age".

For additional information on rule writing in LIPP, go to Appendix 10, on the Syntax of LAL.

10.9 - Multiple file analysis: (UL/LL)

To analyze several files in a row using the same rule set and the same choices regarding Output, one can first go into the Compiler & Editor (or into most wordprocessing programs in a non-document mode) and create a list of file names to be analyzed (there should be a <Return> at the end of each filename and no blanks). Then save the list file under a name you will use for the analysis, say for example Multi.LST (the .LST extension is expected for list files). When you run Analysis from the Main menu, choose "Analyze: Multi File List", pick your multfile list from the menu (of .LST files), and all the files will be analyzed sequentially. Each file's results will be maintained in the order in which they were entered into the list, and at the end of the grand results file, there will be means and standard deviations calculated for each rule outcome (C, O and P) across the various transcription files. There will also be descriptive statistics computed for the calculation metrics included in each analysis.

Multiple file analysis can also be conducted conveniently using lists that are generated

within a special module called "Generate Multi File List" under the "Analysis" option in the Main menu (an innovation of version 1.40). Using this option you can make reference to files in terms of demographic information. For example, you could specify that you want a list of files where all the children are of a particular age, language background, and disability. Assuming the demographics information has been appropriately stored (and everything has been spelled accurately according to your conventions of entry), you could, for example, ask for a list of files where the age of the subject is between 16 months and 24 months (actually the values are entered in days), where the subjects are communication disordered, and where the language at home is Spanish. Then by selecting "Generate File List" from the bottom of the screen, one can initiate the search through the computer's database to find files fitting that description. The lists thus generated are by default stored in a file with a .LST extension and can be based upon Database number, Name, Local ID, Age, corrected age, Languages, Disabilities, Sex, or the Session Comment (which we often use to represent special information in a particular study). After you have generated a list, you select "Display List" and LIPP will indicate the files found that meet the parameter definitions you entered.

The syntax of generating multiple file lists includes the possibility to referencing more than one possibility within some demographics slots. For example, under the slot "Language" in the multiple file list screen, the entry "English, Spanish" (where the comma is important) means "English files and Spanish files". Thus you can specify that your list will include all files where the speaker's language background is one language plus all the files where the speaker's language is another one.

Whatever cluster of parameters one sets up can be saved using the "Save Parameters" option, and if a cluster has been saved, it can be retrieved by using the "Load Parameters" option. Using the Generate Multi File Lists option, it is possible to generate lists quickly for extensive Multiple File Analyses that can be conducted while one goes to lunch.

The output of multiple file analysis includes means and standard deviations for all rule outcomes, percent (P), occurrence (O), count (C), and for calculation rule results. The calculation rule outcomes are also computed based on "grand totals" from the O, C and P values across all the files. In addition the output indicates the outcome for each individual transcript file in the order in which they appear on the .LST file.

11 - Demographics in LIPP (ALL)

The Demographics option in the Main menu can be used directly to enter data on subjects or clients regarding Name, Local ID number (often the Social Security Number), Birthdate etc. Select "Demographics" and you will see the various options. To enter data under any category, simply position the red cursor line at the category of interest and hit <Enter>. Then type in the requisite information. The information stored while one works in the Demographics module goes into a file that is tied to the alphabet selected under "Settings". Thus if your alphabet is set to FATTLIPP, the demographics information you store will be in a file called FATTLIPP.DMG.

In order to view the information associated with any subject, it is most convenient to use the Load Client option to display a list of currently entered clients. It is very important that the information in the client demographic database correspond with that in the headers on files -- so if you call up a particular client from the Demographics option and modify some of the demographic information, you may need to go to all the transcription files that include data on that individual in order to modify them to match. When you generate a Multi File List for database analysis, LIPP will reference the information in the client database, not the information in the transcription file headers. The hard connection between the two is the client number. Thus if you have data in a transcription file, designated by demographics as pertaining to client number x, then regardless of what your header says, the database analysis system will assume that the information in the client database about client number x is the information that is relevant.

Printing demographic information is as simple as selecting a client number and then selecting the "Print Demographics" option at the bottom of the Demographics screen. Demographics information from the file header is also automatically included when transcription files are printed from the Transcription module HELP menu under "Print Document Demographics".

12. LIPP Symbology Development

(1) Creating symbols in LIPP

UL

Most people will probably be satisfied to use the IPA-style symbols in FATTLIPP or TITELIPP. But if you want to make up your own symbols for any reason (like for example that you don't like the recent revisions of the symbology by the International Phonetics Association), then you can do it in the "Edit Characters" option that can be accessed under "Characters" in the MAIN menu.

(a) Making a scratch alphabet file. Before you enter "Edit Characters" however, it may be wise to create an alphabet that you can play with and mess up, without having it do any harm. Do this by selecting the last option, "New Alphabet from Current". When you make this selection, you will be asked to give the name of the alphabet you wish to create. Let's use the letters "scratch" and do not add an extension. Then when you hit <Enter>, all the files you need to create your own alphabet from a copy of TITELIPP will be available as files with the name "Scratch.xxx", (where xxx represents any one of several extensions associated with the various alphabet files), and you won't have changed a thing in the original.

Now go to "Settings" in the MAIN menu. The alphabet you are "set" on will be indicated when you select "Settings". If no one has made any special adjustments, it will be either TITELIPP.alp or FATTLIPP.alp. Position the cursor there using the "down arrow" key and hit <Enter>. A menu of options will appear, and it should include the "Scratch" file alphabet you just created (SCRATCH.alp). Select it by positioning the cursor over it and hitting <Enter>. The name of that file (SCRATCH.alp) should now appear just below the word "Settings" on the MAIN menu line. Now you are ready to mess around without doing any harm to your alphabets.

ONE NOTE OF CAUTION: just about the most common error we have found in using LIPP is working in the wrong alphabet. When you create a scratch alphabet, or change to some new alphabet (not the one you intend to use when you are transcribing), you may forget to reset to the preferred alphabet when you go back to transcribing. Then you may find that some symbols are no longer available to you, or they are in the wrong location on the keyboard, or they don't have the right meanings, or (if they are diacritics) they are appearing in the wrong spaces. ALWAYS REMEMBER WHEN YOU FINISH WORKING WITH ALPHABETS TO RESET YOUR PREFERRED ALPHABET BEFORE GOING BACK TO REAL TRANSCRIPTION.

You need to have a mouse installed when you enter Edit Characters, or else the program will not run. You can however hit <Esc> to get out of Edit Characters if the mouse is not installed, and that way you will not have to reboot. Within Edit Characters, most commands have to be delivered from the mouse.

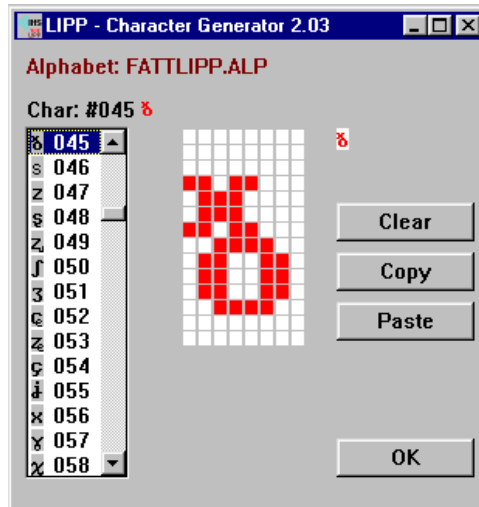


Figure 12.1 – Character Generation Module

(b) Entering "Edit Symbols". Go into "Edit Symbols" (Symbols Menu in the Main Module) and you will see a brand new screen type appear. On the left will be a large eight by fourteen grid, which will be the primary location of symbol generation (the "workspace grid"). On the right will be menus that are accessible with the mouse. On the lower right, you will see a smaller rectangle that is the location where symbols appear at the size they actually will be in transcription. We call this smaller rectangle the "viewing space". As you create symbols, in the big workspace grid, you will see results in the small rectangle or viewing space to give you a sense of what the character you are creating will look like in use.

(c) Viewing symbols. Using the scroll bar to the left side of the work space, you can select individual symbols to work on. You will notice that the numerical indicator of the symbol will be displayed next to the selected symbol on top of the scroll bar. The first 13 spaces (of the 255 slots) are unavailable to LIPP character generation, because they are used by the IBM system. Also 14 to 19 are empty at present (though they can be filled if we wish). But starting at the 20th slot symbols begin to appear. Advance until you get to the 20th character, which will be a [p]. Advance to 21 and you will see a [b]. Note that each character appears in very large size in the workspace grid, but the same symbol appears at just the size you have seen on the screen when you were transcribing in the smaller viewing space to the lower right.

Your TITELIPP symbol list in Appendix 3 will show you all the symbols that Kim made up and their ordinal locations in TITELIPP. These are of course the symbols you are now seeing, because you made a copy of TITELIPP, now called "Scratch", and are viewing it. If you want to scroll through the entire list you can just keep pressing the scroll bar buttons.

Now is a good time for you to simply mess around a little, moving from place to place in your Scratch alphabet, and seeing what the various characters look like in various locations. Each diacritic was designed for a specific location, so you may find that unless you place each diacritic in the right

place, you'll end up with combinations that don't look too good.

(d) Making and modifying symbols. Now it's time to make your own symbol. Select a blank symbol from the scroll bar. This should give you a blank workspace grid. Now move your mouse arrow to the workspace and position the arrow over the upper leftmost square. Press the left button. The square will be filled. Now move the arrow to the next square and press the left button again. The next square will be filled. Now move to the third square and fill it. Fill the whole top line.

Notice that as you fill the line, the image of your creation also appears in the viewing space.

To erase the squares you filled, simply position the mouse arrow over each one in turn and press the right button.

You are now in a position to create any symbol you want that is consistent with an 8x14 grid. Go to it. You will find that it takes practice to get good at making symbols, but if you stay with it you can make very satisfying ones.

You are of course in no danger if you decide to modify the symbols in the Scratch file, since you can return to one of Kim's alphabets whenever you choose to by selecting the "Settings" option in the MAIN menu.

(e) Clearing a symbol. If you think the symbol you are working on (in the workspace grid) has gotten out of hand, and you just want to get rid of it, select the "Clear Character" option in the menu, and the whole thing will disappear. (You can't use this option to clear characters from the surrounding grids, the ones colored blue -- there you have to enter a blank space [such as number 14] under the Disp Char options). If you once create a symbol you like, it may be wise not to clear it without first making a copy, since Kim has learned through experience that you can easily forget how you made up a symbol and it may be difficult to recreate it.

(f) Exiting Edit Characters. When you leave Edit Characters, all the changes you have made in the alphabet will be automatically saved. That's why we don't want you to modify the preset alphabets. If you want to make your own alphabet, that's good, but it's wise to make scratch copies to work on so that you don't lose the originals.

To exit, simply select the "Save & Exit" option on the mouse menu.

(2) Keyboard mapping in LIPP

UL

Be sure your Scratch alphabet is still selected under "Settings", before you start this section on another convenient feature of LIPP symbology, keyboard mapping. With this feature you can put any symbol available in your alphabet in almost any location on your keyboard.

Select the "Edit Keyboard" option from the "Characters" option in the MAIN menu. As with the

"Edit Characters" option, the mouse will need to be installed to work in keyboard mapping.

The screen that will appear will represent the keys on your IBM computer, and the TITELIPP symbols that have been placed in each of the key locations at the top part (or "mapping area") of the screen. The bottom of the screen offers the menu options for your mouse.

In the mapping area, each little column represents a single key, and includes first the unshifted symbol for the key in question in a standard IBM keyboard -- the symbols here are in blue. Below the blue standard symbols there are four little yellow rectangles per column. The top rectangle in each column will be filled with the (white) LIPP symbol that Kim put in that location, the "unshifted" LIPP symbol for that key. The next square down will include (if there is one) the symbol for the <Shift> LIPP symbol at that key, the next the <Ctrl> symbol, and the bottom square, the <Alt> symbol. Blank spaces simply mean that the locations are open and currently unused in the alphabet selected.

For example, find the "E" key on the mapping area. Look for the blue E between W and R. The top symbol "E" represents the standard IBM symbol. Below it, the four symbols represent respectively 1) the mid front unrounded vowel [e] 2) the mid lax (somewhat lower) front rounded vowel, represented by the capital greek epsilon, 3) the mid front rounded [oe], and 4) the mid lax (somewhat lower) rounded vowel [OE].

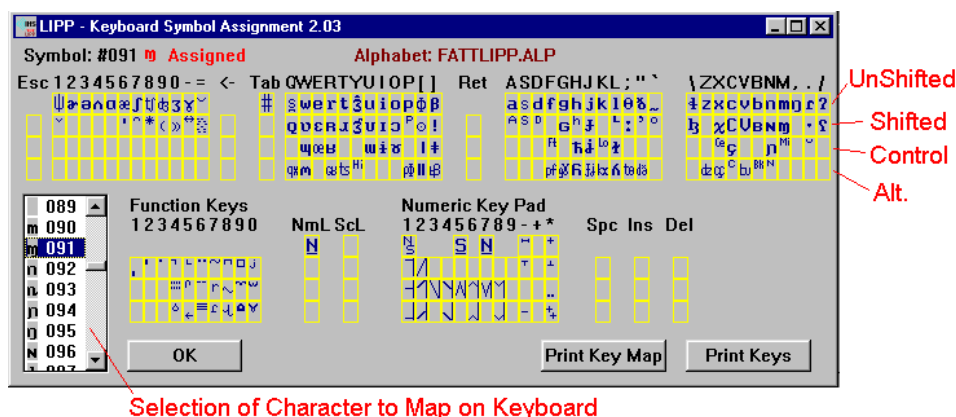


Figure 12.2 – Symbol keyboard mapping module

(a) Removing symbols from the keyboard. Find the "P", on the right hand side of the same grouping that has the E. Note that the orders and groupings represent the lines and orders of symbols on a standard IBM keyboard. Now move the mouse arrow to the spot of the top square, for the "p" key, the one representing the unshifted value for that key in LIPP, the symbol [p] in white. Make sure the arrow is precisely pointing to the white [p]. Hit the right button and the [p] should disappear. If it doesn't disappear (or if some other symbol does disappear), then you didn't have the arrow positioned exactly right. There's no harm if you knocked out the wrong symbol (you can fix it later, and you're working in a scratch file anyway). Keep working until you knock out the [p]. A moment ago, symbol number 20, the [p] was mapped to the unshifted position for the key "p". Now

there is no [p] on the keyboard. If you were transcribing at this moment, using the SCRATCH.alp alphabet, you would find that hitting the unshifted "p" key would yield no symbol at all, because the symbol is "not assigned". The symbol still exists in the alphabet in location 20, however, and can be replaced in any location you choose.

(b) Stepping through the symbol list. Move the mouse arrow to the menu at the bottom of the page and select the option "Char. (character) selection", by positioning the arrow over the blue square, and pressing the left button to step through the available characters. The first one you come to should be the [p], the one you just deleted from the keyboard. Notice that the [p] is listed as "not assigned". Go forward another space by hitting the left button again, and [b] will appear. Note that unlike [p], the [b] symbol is still "assigned". Press the right button to go back to the [p].

You can step through the symbol list using the right button and left button. The symbols are all there in the same order that you find them in Appendix 3.

(c) Placing symbols on the keyboard. Now let's put the [p] back where it belongs on the keyboard. While you are at location 20 on the menu at the bottom of the page, move the mouse arrow back to the square in the mapping area where the [p] symbol was removed a few moments ago. With the arrow pointing precisely to that spot, press the left button and the [p] should be replaced.

If it doesn't work the first time, try repositioning the arrow just a smidgen. Keep working until you get the [p] back in that top square.

Now move the arrow to an open square anywhere on the keyboard. For example, you could use the third square in the same column, the one representing "control p". Position the arrow precisely over that empty square and press the left button. The [p] symbol should now appear in two places in the mapping area, the unshifted square for "p", and the control square. Note also a small capital "P" that appears in the shifted square. That's a diacritic (character 231) that means "unspecified plosive", and occurs preferably in a superscript position (right) when you are transcribing.

You can put [p] in any empty square on the keyboard. Each time you do it, you create an option for the transcriber to type his/her [p] symbol by hitting another key. If the keyboard mapping includes any symbol in multiple locations, then the different locations will all yield the same symbol (and the same phonetic meaning to the analysis system).

(d) Fast movement in keyboard mapping. You can move around fast in keyboard mapping by using the "Goto Char." option at the bottom right. This option allows you to select a character number and go there immediately.

Now it's time to play. You have a scratch alphabet. Make some changes in the keyboard using the procedure indicated above. When you are done, exit by selecting the exit feature at the bottom left of the screen. Then you can (without resetting alphabets) go into transcription and try out your changes.

(e) Enabling symbols for FATTLIPP. If you want to use broad transcription most of the time, but there are just a few symbols in TITELIPP that you wish to have available to you, you can enable them easily. All you have to do is to go into the FATTLIPP alphabet (by selecting FATTLIPP.alp under "Settings" in the MAIN menu), and then entering the "Edit Keyboard" option under "Characters" and assigning the symbols that you want to keyboard locations you choose. You see the missing symbols are actually present in FATTLIPP, they are simply unassigned until you decide to assign them. When you're finished with assignment of the keyboard, following the procedures indicated in the two sections above, you can exit and go to "Printing new keycaps." You may also wish then to make adjustments in symbol menus consistent with the changes, depending on what you have done previously (see below on "Menu adjustments").

(3) Printing new keycap covers

UL

If you have created a new keyboard mapping that you wish to use, you will want to redo your keycaps in keeping with the changes. This is easy. While you have your new alphabet selected under "Settings" in the MAIN menu, simply go to "Characters" and select "Print Keyboard Caps". Assuming your printer is ready, you will get a two page printout (this takes several minutes to run, so be patient) of keycaps indicated by your keyboard files selected under "Settings".

Now you need to cut out the keycap covers with a pair of scissors and replace the covers that correspond to changes you made on your keyboard.

(4) Redefining symbols

UL

The phonetic meaning of a symbol in LIPP is determined by the values the symbol has on the 16 dimensions. Readjusting those meanings is possible, but you should be sure to think through the implications of doing it, before you make permanent changes in TITELIPP or FATTLIPP. The rules you use depend on the meanings assigned. Kim has written lots of rules for you and if you change the meanings of symbols, you run the risk of foiling the analysis rules that are set up for you.

This is a good reason for continuing to work in a scratch alphabet while you are learning to use alphabet definition controls. So be sure you have selected a scratch alphabet under "Settings" before you start the next exercise.

Under "Characters" in the MAIN menu, select the "Edit Character Definitions" option. A new screen will appear, that will show you a top line in red and then 16 digits in two rows of eight (they will all be 0's at this point). Below the two rows of eight digits are indicators of "position" that the symbol in question is required to assume (main symbol, or if it's a diacritic, top, top right, right, bottom right, bottom).

The top line on the screen indicates procedures. It reminds you that "up and down arrow" keys can be used to step through the symbol list (character selection). The "right and left arrow" keys will

be used to select changes in the values on the 16 dimensions (parameters), or the position indicators. <Esc> is used to exit and save the changes in the character definitions.

(a) Stepping through the symbol list. Use the "down arrow" key to step through the symbols. Press it repeatedly and you will see the various symbols in the normal order appear in the upper left segment of the screen. If you hold the "down arrow" key down, you will see the system advance rapidly through the symbols (and blank spaces among the 240 available ones).

Using the "up arrow" key, you can go backwards in the symbol list. Go back to location 20, the beginning of the list, symbol [p], and stop there.

(b) Stepping through the dimension values for a symbol. Look at the 16 dimensions. In the upper left hand corner of the two rows, you will see a "1" representing the fact that [p] has a "1" on the first or A dimension. It has a 0 on the B dimension and a "1" on the C, etc.

Press the "right arrow" key and you will see a red cursor appear over the A dimension digit. Also you will see the screen light up with a green box offering possible changes in the dimension (or "parameter") values. In white at the top of the green box the letter designation (in this case "A") of the parameter is also indicated. Press the right arrow again and the red cursor moves to B, then to C, and so on. The position of the red line in the green box indicates the value that has been set for the character in question on the parameter in question.

Keep stepping through the parameters and eventually you will step to the "position" locations, where [p] is indicated to be a "main" symbol both in the primary and secondary positions.

If you step past the last option (secondary position), you will return the cursor to the top of the screen where you can then continue stepping through the symbol list. While the cursor is on the two rows of digits or the position spaces, it is not possible to advance through the symbol list (the "up and down arrow" keys have a different function now). You have to get the cursor back to the top by using the "right or left arrow" key, or by hitting an <Esc>. (Be sure you just hit one <Esc>, because two will take you back to the MAIN menu).

Stay at [p] for a few more moments.

(c) Examining the meanings associated with the dimensions. Use the "right arrow" and step to the A dimension, the upper left hand digit (a "1"). Now hit F1 and you will see an indicator that shows the meaning of the A dimension. Note that the value "1" on that dimension is "plosive", the value that is given to [p]. Using the "right arrow" key you can continue stepping through the dimensions looking at the meaning lists for each one.

Note that the meaning lists look just like the ones in Appendix 2. Hit one <Esc> to make the dimension meaning lists disappear.

(d) Changing symbol meanings. Now return to the A dimension and hit the "down arrow" key several times until your cursor is positioned over the value "7" (which means "nasal"). Hit <Enter> and the value of the A dimension on the [p] changes from "1" to "7". At this moment in time, you have defined [p] as a voiceless bilabial nasal. If you left the "Edit Character Definitions" section of the program now and ran an analysis, the [p] symbol would be interpreted as a nasal. Not a good idea, but the example hopefully clarifies the importance of the meaning lists.

So move your cursor back to "1" using the "up arrow" key and hit <Enter>. This will restore the correct meaning to the [p].

(e) Position adjustments. Using the right arrow key, step through until you reach the primary or "1st position" slot. Notice that the options available in the green box are the various locations in the symbol grids that you can place symbols. [p] is indicated as a "main symbol" in both the primary and secondary locations.

If you select one of the other position options using the "down arrow or up arrow" key, and then hit a <Enter>, you will change the value of the symbol in terms of its placement in the symbol grids. Be sure when you leave [p], it has the values "main" for both primary and secondary positions.

Step through symbols rapidly using the "down arrow" key until you reach the diacritic sections. Select a diacritic to look at, say number 171, the dentalization symbol. It has an "nc" or "no change" value on all the dimensions except the B dimension which has to do with tongue position. There its value is "1", meaning dental. Hit F1 to look at the meaning of the B dimension and verify it. Hit <Esc> to return to the previous mode. Notice that the dentalization symbol is indicated for a primary position of "bottom" and a secondary of "bottom right". The choice of the primary position is in keeping with the IPA preference for the symbol. The secondary position is chosen by Kim to be the next best location in the event the primary one is filled.

(f) Changing dimension meanings. Using the right arrow or left arrow key, step to dimension C "lip posture" and hit F1 to look at the meaning list. You will see the values that Kim assigned to lip posture. These he chose to incorporate as much information as possible from the IPA and to leave open additional options for phonetic description of child and infant vocal sounds. Kim made these choices with much agonizing and meditation, so it would be unwise to change them, except in these scratch alphabets, unless you have a very good reason for it. Remember that the analysis systems will be affected by the choices made here.

But since we are in a scratch alphabet, we can change these meanings if we want. Using the "down arrow" key you can select any row hit <Enter> and then modify any of the terms on the list. Or you can select any of the blank spaces and enter some new meaning.

The information you are entering when you write in these dimension meaning spaces are mere names. They don't directly change the way anything else functions. But once you make these kinds of changes, they may affect how you wish to select dimensional values for particular symbols, so

that the symbols will be categorized according to your new terminology. The dimensional meanings are the terms that guide how you categorize (i.e., code in the 16 dimensions) new symbols if you create them, or redefine old symbols if you have extremely good reasons for thinking Kim's definitions don't suit you.

When you are finished writing in dimension meaning spaces and moving about the alphabet listing, hit a couple of <Esc>s and you will return to the MAIN menu. When you do that, all the changes you made in the symbol meanings and dimension meanings will be saved under your scratch alphabet name.

(5) Selecting diacritic combinations

UL

In order to keep transcribers from entering illogical combinations of symbols, LIPP offers the opportunity for the user to set a variety of limitations on how symbols can be combined in the grids while transcribing. For example, a [p] is a consonant, and tongue position is considered irrelevant. Thus it would be unuseful to mark a [p] as having a "raised" tongue position (symbol number 162). Consequently, Kim decided to specify that an error message would appear anytime a user tries to mark a [p] as "raised" using symbol 162. Furthermore, the diacritic that means "lowered" (symbol number 163) also refers to tongue height of vowels, and would have no meaning if associated with [p]. Kim disallows that combination as well.

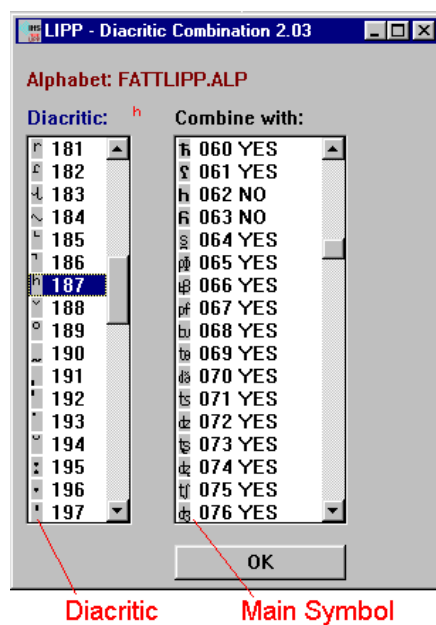


Figure 2.3 – Diacritic Combination Screen.

In order to manipulate the allowability pattern of any symbols, you select under "Characters" the option "Edit Diacritic Combinations". Be sure you are still in your scratch alphabet while you do this. When you have selected "Edit Diacritic Combinations", you will see a new screen appear. At the top of the page you will see procedural instructions and an indicator about your location in the

symbol list. Only potential diacritics will appear here (i.e., symbols that have been defined as having at least one position, either primary or secondary, that is not a "main" position). Using the "right and left arrow" keys you can scroll through the list of diacritic symbols appearing on the left side of the red line at the top of the screen.

At the right lower side of the screen, as you scroll with "right and left arrow" keys, you will see the symbol list that includes everything in the alphabet, and to the right of each symbol the word "yes" or "no". A "yes" means that the diacritic symbol in the upper left of the screen is allowed to combine with the symbol next to the "yes". A "no" means the combination is disallowed.

(a) Changing diacritic combinations. To change a diacritic combination, select a particular diacritic using the scroll bar on the left side of the screen (list showing diacritics). When you find a diacritic you are interested in, click on the diacritic. This will update the right side of the screen, showing which main symbols can be used with the selected diacritic. Let's try the aspiration symbol, number 187 – click on the diacritic. Now use the up and down scroll bars on the right side of the screen (list showing main symbols) to move down the list of all the alphabet symbols. Notice that the aspiration symbol is allowed with nonglottal consonants in general (even voiced consonants can be aspirated in Indic languages), but that vowels get "no" on aspiration.

If you disagree with Kim about any of the combinations of aspiration and main symbols, you can change the value for any of the main symbols by left mouse clicking on that symbol. If the value was "no", you change it to "yes" with a click of the mouse and vice versa. Try it.

Go down the list of symbols. Notice that the aspiration symbol is not allowed with some other diacritics, for example, the symbol for vowel raising, number 162.

(6) Adjustments in symbol menus

UL

The symbol menus that you see when you are in Transcription are from a file that is named in accordance with the alphabet the menu corresponds to. For example, TITELIPP.alp has a menu called TITELIPP.men. Any file with a .MEN extension is a menu file. At this moment there should be a menu file called Scratch2.men.

To change your menu you need to edit the menu file. There are two ways to do that. One is to exit LIPP and go into a wordprocessing system where you can edit the file (Scratch2.men). Another is to go to "Analysis" under the MAIN menu, select the "Enter Compiler & Editor", go to "Files", and select it. Select "Load Program File" and hit <Enter> when the list of files comes up and then select a new path by entering *.men <Enter> and you will see the list of menu files that are available. Select the one you wish to modify (the scratch.men file).

Then when you go to Edit and hit <Enter> you will be in the LIPP editor ready to change the menu file.

Kim provided two identical menu files, TITELIPP.men and FATTLIPP.men. He thought that those of you who use a broad alphabet might like to keep all the narrow symbols visible in menus just in case you want to consider more narrow options at some point. But if you hate having unused options appearing in your FATTLIPP symbol menus, you can delete them by editing either in a word processing system or in the editor of LIPP. (Remember that to delete a line in the LIPP editor you hit <Ctrl> page up).

Uncompiled symbol menus are organized in sections that begin with four @ symbols (to indicate beginning of a "consonant", "vowel" or "diacritic" menu -- the only major options), and first level subheadings (which will be displayed horizontally in the compiled menu) that start with a single @ symbol followed by the first level subheading name (user adjustable). On subsequent lines of the menu file, second level subheadings (which will be displayed vertically under the first level subheading in the compiled menu), begin with three zeros followed by the second level subheading name (user adjustable) that will appear in the compiled menu. Individual menu entries under each second level subheading begin with the three digit numerical value of the symbol location (014 - 255, as indicated in the "Print Character Definitions" file, Appendix 3), followed by a user adjustable description of the symbol's meaning, followed by a keyboard position designation.

LIPP automatically compiles the .MEN file that is relevant at run time. Compiled menus will display characters rather than their numerical indicators against a light blue background, first level subheadings on the top line against a black background, and second level subheading titles highlighted in white arranged vertically under the appropriate first level subheading.

When you are finished editing your menus, exit the wordprocessing system or editor (be sure to save the file if you want the changes to be kept).

(7) Defining new alphabets in general

UL

If you feel you would like a completely new alphabet, it is wise to learn first how to manipulate the various parts of the LIPP system under "Characters". It is also wise to understand LIPP analysis thoroughly. You don't want to start creating a new alphabet willy nilly. You need a plan, because the decisions you make early on will affect how easy it is to complete your system and create appropriate analysis capabilities.

Some kinds of changes are much less influential in the system in general than others. For example, you can change keyboard locations of symbols quite freely without affecting meanings or analysis at all. You can also change menu options with little effect. But if you take a symbol that Kim has in one location (say [p] at location 20), and put it in a new location, say location 130, you will need to redefine elements in those locations, and it may require some thinking to get it right.

If you choose to redefine the 16-dimension code, you will need to redefine symbols in terms of

that code and redo variable, global and rule definitions in analysis in terms of all the new dimension and symbol definitions. And you will need to create new menus.

Don't get us wrong. You can create your own alphabet. LIPP was created with the expressed intention of making that possible. But when you redefine things, you have to think about the logical implications of your decisions.

13. Additional Features of LIPP

The following section in italics is not currently available in the Windows version:

(1) Creating Secondary Indices or Dictionaries **ALL**

A secondary index is a dictionary that is created first by the same means that regular dictionaries are created and then modified in terms of ordering in the LIPP editor (or a wordprocessing system). The advantage of this is to offer dictionaries (accessed through the "Search by Secondary Index" option of the Dictionary menu) that do not have to be ordered alphabetically, but can instead be ordered by feature groupings, or number of syllables -- whatever you like. While editing, you mark lines that are to be taken as mere labels with three zeros at the beginning and then what comes after will be understood by LIPP as a labelling sequence and not dictionary entries. You can have lots of different segments in your secondary index.

You can also create different indices for different functions. You need however to set the "Second Index:" function in the Dictionary menu to the index you wish to access at any point in time. Then "Search by Secondary Index" will put you in the index you want, and some of the coded information, such as the zeroes and the numerical values that go with the vocabulary items will not be seen. You need to preserve those numerals though in the editing process, because LIPP uses that information for its own purposes.

When you "Add Word to Dictionary" from the Dictionary menu, the new word will go in the main dictionary that is set under "Dictionary" (the top line in the Dictionary menu), and in the secondary index (set under "Second Index"). It will go at the end of your secondary index, and if you wish to reorder it, you enter your editor or the LIPP editor to do so. It is not acceptable to add words to your secondary index by typing them in, because you may not know the numerical values that LIPP needs to manage the index. If you enter them through the Add Word to Dictionary feature, however, the numerals are put in for you.

(2) Using the Special Transcript Line Features **ALL**

There are a number of nonphonetic ways that the LIPP transcriber can code his/her tape or vocal session while transcribing it phonetically. For example, the tape counter feature allows you to indicate with a number, the location of each utterance on the tape you are transcribing. Even more detailed information about location can be entered using the timing features which allow you to enter word-by-word information (or segment by segment information) regarding the duration of sequences of transcript or silence. Finally LIPP allows you to enter a four character code under "space information" that can code whatever meanings you want (at the University of Miami the team uses it for a pragmatic/semantic coding), or a long message for each Line of transcript under "Line comment" that can be your comment about the data entered on that Line.

(a) The tape counter feature. For most research transcribers, tape counter numbers are a critically important aspect of a transcript because they facilitate locating the utterance in question for reevaluation or for spectrographic analysis. Use the “right” mouse key, to pull up the tape counter information dialog box.

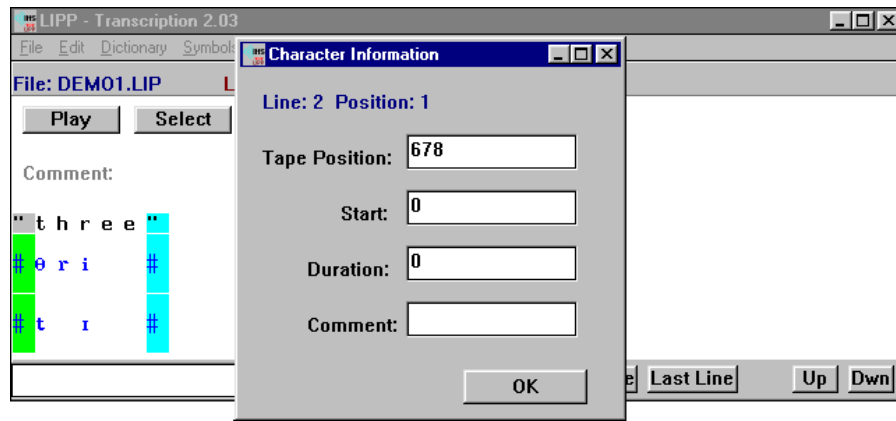


Figure 13.1 – Character Information dialog box showing input options for Tape Position, Start (Utterance – Tape Position), Duration (of Utterance) and Comment.

In general, you want to enter your tape counter number while your transcription cursor is at the very beginning of the Gloss, Target Form or Transcription Row (i.e., in Position 1). This is the counter number that LIPP will refer to when merging files.

LIPP allows you to enter tape counter numbers in any Position (i.e., column), to indicate the counter numbers corresponding to different syllables or words on the Line. The counter number you enter at a particular Position will appear on the screen at the top in the middle of the light blue bar, only when the cursor is at that Position, except for the first Position counter number, which once entered, will stay on the screen at the far left no matter where the cursor is on the Line.

LIPP will protect you against entering nonsensical tape counter numbers. For example it will stop you if you have entered numbers out of order (i.e., row 1 at counter 15 and row 2 at counter 13), requiring you to correct the mistake. That way you don't easily make counter number errors by simply mistyping.

You can go into the Transcription module and look at a file with tape counters entered if you like. It's called DEMO2.LIP, and there are counter numbers at more than one locations on some Rows. Note for example on the first Row of the sample, the counter number is 21.0, and at Position 5 (move the red cursor to that Position to see it) there is an additional counter number, 21.2.

(b) The timing feature. Like the tape counter feature, LIPP allows the transcriber to enter information about duration of elements in a transcribed string, Position-by-Position by clicking on the right mouse key at each transcription position.

The “Start” entry is a value that the user decides by reference to whatever standard s/he wishes at the beginning of each Line (the best choice may in general be a 0). But thereafter on the Line, the value needs to be logically consistent. If at Position 1 you enter a start time of 50, then at Position 10, if the element in question occurs 350 ms later than the element at Position 1, the start time entered should be 50 + 350 or 400. The values entered are interpreted to be in msec (integer values only).

The second entry, “Duration” corresponds to the value of the duration of the segment (or segments) following the designated start time. For example, if the Transcription Line in question has four words with pauses between them, you may wish to enter start times at the beginning of each word, durations for each word, and at some future date when Rafael creates an analysis of duration data, we will be able to calculate the silence durations between the words from the start times and durations entered.

If you plan to do duration studies on complex discourse, the timing feature may be very useful. In DEMO2.LIP, timing information is entered on the first Line. Move the cursor to Position 5 (the y on “crayons”) and you will see timing indicators, 100 for start time, and 155 for duration.

(c) The Comment feature. Space Comment information is whatever you want it to be in four characters. We use it for pragmatic/semantic coding. For example in DEMO2.LIP, the first Line, if you position the cursor at Position 1, you will see on the far right of the light blue bar, in quotation marks the letters “IMIT”, meaning to us “imitation”.

(d) Line comments. Line comments are also whatever you want them to be. You can enter a line comment by clicking on the gray “Comment” label above the Target line at any time.

(3) File merge

UL/LL

When you merge files to do reliability analysis (accessed from the “Other” option in the Main Menu), select File Merge and a new menu appears. The red bar indicates the location of your choices. The first option is “Row to Use” and it can assume the value of Transcription row or Target Form row (referring to rows in the transcription files you wish to compare after merging). In general, a reliability analysis compares two Transcription rows (not Target Form rows), so you will probably always choose Transcription here (<Enter> changes the value from Transcription row to Target Form row).

Use the down arrow to move the red selection bar down to the second option “Target Form”. The choice here, selected by hitting <Enter> and then entering a transcription file name, determines which of the two transcriptions to be compared for reliability will be the standard (let’s call it transcription 1) of comparison in the merged file -- transcription 1 will appear on the Target Form row of the output or merged file. The next option, “Transcription row” is selected the same way, and you enter the filename of transcription 2, the one you wish to compare with transcription 1.

The Output file name which you enter determines the name of the merged file, the one you will later analyze to assess the degree of similarity of transcriptions 1 and 2.

When you select "Perform File Merge" the merging will begin (assuming you entered correct file names). If all the transcribed speech segments in transcriptions 1 and 2 are in the same Line and Position locations, File Merge will align everything perfectly and you will be ready to perform an analysis. You can check the alignment situation very quickly by stepping through the merged file, Line by Line using the "down arrow" and "up arrow" keys. The Target Form file (transcription 1) is on the left, and will appear Line by Line, main symbols only (no diacritics), as you scroll down with arrow keys. The Transcription file (transcription 2) is on the right. The merged output is at the top, showing the row from transcription 1 in the Target form location, and the row from transcription 2 in the Transcription location.

If, however, your original files were not aligned in just the same ways, Line by Line and Position by Position, then there will be some editing to do. Editing is especially important when one transcriber has included utterances that the other transcriber has left out. As a result, transcription 1 may have utterance x at Line 12 while transcription 2 has it at Line 13. In that case you'll want to move the contents of one of the Lines to match them up in the merged file (for example, Line 12 of transcription 1 could be moved to Line 13 of the merged file where it would be matched up with Line 13 of transcription 2). The File Merge editor makes this and other simple editing tasks much faster than they would be otherwise. One can always do the editing of the merged file by exiting File Merge (by hitting <Esc>, <Esc>), entering the Transcription module and loading the merged file. However, within the Transcription module, such operations as moving a portion of one Line to another Line are cumbersome. The File merge editor is specifically set up to handle them.

In white you will see counter numbers if they were entered at the time of transcription; these, along with the Gloss rows may help you decide which Line from transcription 1 goes with which Line from transcription 2. The side of the screen that is lit up in red is the active side, and the side in green is inactive. You can change the active side with the "right arrow" or "left arrow" key. Note that when the left side is red, the Target Form row in the merged file is red.

When you have decided that Lines are unmatched, there are several functions that can be recruited in File Merge to get them properly aligned in the merged file. If you hit <Delete> or <Insert> while on any Line, you will change the red lit file in accordance with your selection. For example, if one file has an extra utterance in it (one that the second transcriber simply did not include in his/her transcript), you can delete that utterance by hitting <Delete> while it is lit up in red. This automatically adjusts the numbering so that the following transcriptions are brought up to fill in the spot left open by the deletion. If you hit <Insert> while a particular Line is lit up, it will insert a new (empty) Line in that location. (Once you delete a Line, you can't get it back except by going back into Transcription, so don't do it unless you are sure).

The menu on the left side of the screen offers editing capabilities while you are merging files. If

one Line of transcript includes words or sequences on a previous or succeeding Line in the other transcript you can go into the File Merge editor by hitting the <space bar> and then selecting the spot from which you wish to "Break Line Down" (move the portion of the Line beyond the cursor to next Line down) or "Move Line Up" (move the portion of the Line beyond the cursor to the previous Line). When you break a Line down, you send the contents of the break-down to the next Line, all by itself. All subsequent Lines are pushed down by one. If you move up or break down an entire Line, you leave an empty row behind, which can, if appropriate, be refilled by exiting the editor (by hitting <Esc> just once) and using the <Delete> key to bring the contents of subsequent rows up one Line.

While in the File Merge editor, you can edit the Lines in order to get everything squared (remember that lining up is necessary for analysis to work right) by using <Insert> and <Delete> while you are in the edit mode on each Row of the merged transcript. Further you can store any character by hitting S and then retrieve (R) that character and insert it. This is especially useful for placing boundary markers where they belong in the edited transcript. When you are finished with editing your Line, hit <Esc> just once (twice will end the merge session) to get out of the editor and then you can look at additional Lines that might need editing prior to reliability analysis.

Once all the Lines are properly ordered and merged, you exit with <Esc>s, the file will be saved under the name you entered under Output File, and if for any reason you need to, you can edit the file further in Transcription in order to make sure all the word boundaries line up, and all the relevant segments do too, since LIPP analyses, including reliability analyses require lining up.

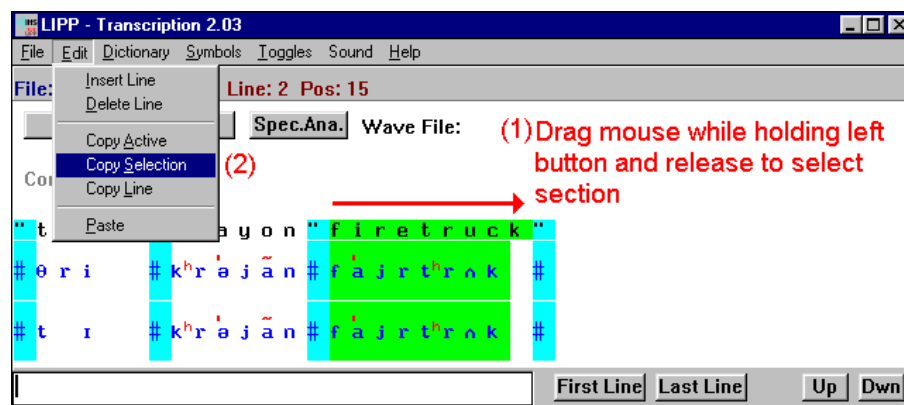


Figure 13.2 – Cut and Paste procedure for copying transcription segments.

(4) Save & Retrieve, Search & Replace **ALL**

While you are transcribing a spontaneous speech sample from a particular child, it is possible you will find the same word being repeated. Rather than using the dictionary to look the Target Form up each time, you have the option of saving the gloss and target in a buffer from which you can quickly retrieve it each time it occurs in the sample.

The save and retrieve functions can be used to edit files where the user wishes to take portions of one Row's transcript (say the last two words of a five word utterance), and put them on a succeeding Row. The whole Line is saved, the Insert Transcription Line feature is chosen from the Main Menu, and the saved Line is retrieved in the newly created Row. Then the delete key can be used to select the portions of the transcript that are desired on each Row.

If you make a consistent error, that is only discovered later, you may wish to change an entire long transcript. For example, suppose you transcribe all the [s]'s in a sample without indicating "blading", with the little square symbol, and then you decide that you really should have included that diacritic on every [s]. Use this feature to correct the entire transcript.

(5) Available alphabets

ALL

While users of UPPER LIPP have the capability of modifying alphabets available in LIPP, and can if they wish create an entirely new alphabet in LIPP, it seems likely that many users of LIPP will adopt one of the IHS supplied alphabets. Consequently, it seems worthwhile to offer some background on the symbologies that have already been created.

The International Phonetics Association published the results of its most recent convention on the IPA alphabet in 1989. In the development of the symbology for TITELIPP and FATTLIPP, we have attempted to adhere to the form of symbols specified in the 1989 publication. There are, however, a few departures from the IPA that deserve comment.

First, the IPA lacks symbols for a number of characteristics of young child speech and infant vocalizations, characteristics that have been described and discussed in published work. We have added a number of special symbols to account for some prominent examples of child articulations. For example, we use the diacritic asterisk to denote a "marginal transition", i.e., a formant transition that is slower than normal, causing the syllable to seem hesitant and ill-formed (a slow releasing transition is indicated by placing the asterisk on the consonant immediately preceding the ill-formed transition, while a slow arresting transition is indicated by placing the asterisk on the vowel preceding the ill-formed transition). A number of authors have talked about "salivary" articulations of infants and young children. Consequently, a diacritic for salivary articulations is included (made up by Kim).

Second, the IPA does not utilize "global symbols". We have chosen to include globals such as P for "plosive", C for "consonant", in order to offer the user the ability to characterize elements in a partial manner. Arlene Carney has dubbed this sort of approach "Reduced Aspect Feature Transcription (RAFT)" and has formulated special ways of transcribing in RAFT. LIPP offers the opportunity for the user to use global transcripts when deemed necessary, and to shift back to more detailed transcripts at other points in time -- even within the same utterance.

Third, the LIPP alphabets include boundary symbols to indicate different levels of phrase or

discourse structure. The symbols we have chosen include the word and utterance boundary which are relatively traditional in child phonology. The word boundary functions as a sort of default boundary in the system, but of course that can be changed in UPPER LIPP. Other available boundaries are intended to be interpreted in the way poetry unit boundaries are interpreted -- the syllable, the foot, the line and the paragraph (or verse) designate units of progressively greater dimensions with definitions that are intended to conform to the poetic tradition and to metrical phonology. The UPPER LIPP user, of course, can define these or other boundaries according to their own goals. For those who plan to modify boundary symbols, it must be noted that boundaries have to be placed in character positions 250 through 255, if they are to have the blue background that LIPP uses to highlight boundaries, and if they are to be interpreted as boundaries by the IHS supplied rule systems.

The fourth difference between the IPA 1989 and the standard LIPP alphabets is that we have taken the liberty of making a few relatively minor adjustments in the symbols in order to adapt them to LIPP screens and in order to avoid unnecessary symbol redundancies (of which there are several in the IPA). The locations of diacritics in LIPP are generally the same as locations specified in the IPA, but there are occasions where two diacritics may be placed on the same main symbol. In such cases LIPP cannot put both symbols in the same location (for example, below the main symbol), even if the IPA would suggest such an arrangement. Consequently, we specify in LIPP alphabets two locations for each diacritic, a preferred one and a secondary one, which is used whenever the preferred location is already filled with a diacritic.

So TITELIPP and FATTLIPP are both basically IPA with some modifications. FATTLIPP is nearly a perfect subset of TITELIPP.

Many symbols available in TITELIPP are not mapped to the keyboard in FATTLIPP (but can be reinstated, in UPPER LIPP, see "Keyboard mapping", above). FATTLIPP also uses the standard typescript "r" for the English retroflex "r" in keeping with current American practice in speech pathology, while TITELIPP uses the upside down version, in keeping with more international practice. TITELIPP uses the typescript "r" for the alveolar trill, and FATTLIPP includes a special "double r" symbol for the same purpose. In FATTLIPP, affricates are typed in a single stroke, and fill a main symbol position only. TITELIPP includes the ability to type affricates with the single stroke, or to type a stop consonant, followed by the homorganic fricative as a diacritic. The advantage of this arrangement is that the affricates are a little prettier, but you should remember in TITELIPP that fricatives are categorized as both main symbols and diacritics, and you may encounter cases where LIPP gives you a message about inappropriate diacritic use when you try to use fricatives.

(6) Page saver printing: Print document compressed **ALL**

Transcript files are sometimes prepared in a way that is wasteful of paper at the time of printing. If you have just one word per Line of transcript, for example, then most of each Line will be blank. To save paper, you can print out your files in a "compressed" mode by selecting "Print document compressed" from the F1 or "Help" menu while you are in "Transcription". This function

automatically prints so as to fill up Lines, indicating the original Line number of the first item on each printed Line. Compressed printing does not, however, split up material that originated on a single Line.

Don't be concerned about wasting disk space by inefficient transcript structure. The LIPP software automatically stores transcripts in a compressed mode.

14. How to Reference LIPP

LIPP can be referenced as an unauthored computer software program. The IPA format is as follows:

LIPP (version 2.10) [Computer Software]. (2001). Miami, FL: Intelligent Hearing Systems.

15. Index

alphabet
Available alphabets, 77
compatibility, 13
Defining
new alphabets, 70
make your own symbols, 60
making a scratch
alphabet, 60
Print Character
Definitions, 35
role in database, 33
warning about
compatibility, 24
warning on compatibility,
13
analysis
Correctness analysis, 38
Displaying results of an
analysis, 40
example of rules, 47
Inventory Analysis, 41
inventory correspondence
analysis, 37
Lexical or phonetic
avoidances, 54
Multiple file analysis,
57
Performing an analysis in
LIPP, 39
phonological processes,
39
Printing results of
analysis, 41
Reliability Analysis, 42
Saving results files, 41
Segmental Inventory
Correspondence analysis,
41
transcription reliability
assessment, 42
column (position), 15
Cursor
automatic advancing, 20
location, column
(position), 15
location, line, 15
location, row, 15
movement in
transcription, 16
movement, line and row,
19
positioning for adding
words to dictionary, 28
positioning for
dictionary search, 27
positioning to type
diacritics, 16
rapid movement line to
line, 25
database
connection to alphabet in
use, 33
demographics
printing demographic
information, 59
Demographics
defining new categories,
32
meaning of entries, 32
Saving data, 32
dictionary
secondary index, 72
FATTLIPP
dictionary, 29
differences from
TITELIPP, 29, 78
enabling symbols from
TITELIPP, 65
warning about
compatibility, 13
Feature correctness
analysis, 38
File merge, 74
Installing LIPP, 8
disk space, 8
Reminder about Placing
the ACTIVATOR, 9
Intertranscriber
reliability, 43
Intratranscriber
reliability, 43
Inventory analysis
(context free), 37
Inventory feature counts
(context sensitive), 38
keyboard
Converting your keyboard
to LIPP, 9

keycaps, 9
 transparent keycaps, 10
 Keyboard mapping in LIPP, 62
 Levels of LIPP, 5
 THIN LIPP, 5
 LEVELS of LIPP
 UPPER LIPP, 5
 Line comments, 74
 Lines, 15
 LOWER LIPP, 5
 Menu adjustments, 69
 Ordering constraints on rules, 54
 Phonetic avoidance analysis, 54
 phonetic symbols
 Definition of phonetic terms, 45
 generating new symbols, 70
 maximum number, 35
 meanings, 35
 the 16 dimension code, 35
 Phonological Process analysis, 39
 Print document compressed, 78
 Printing new keycaps, 65
 Redefining symbols, 65
 reliability analysis calculation metrics, 54
 File Merge, 53
 Report generation buffers, 57
 calculation rules., 56
 Display statement, 55
 If/Then statements, 57
 Report Generation, 55
 running an analysis, 49
 Running LIPP, 12
 Save and retrieve functions, 77
 Scratch alphabet file, 60
 Secondary Indices or Dictionaries, 72
 Selecting diacritic combinations, 68
 Settings menu, 12
 alphabets, 12
 Space information feature, 74
 Symbol menus, adjustments, 69
 Symbology Development, 60
 Tape and timing features tape counter, 73
 timing, 73
 Tape and Timing Features, 72
 TITELIPP differences from FATTLIPP, 29, 78
 warning about compatibility, 13
 Transcription module, 14
 cursor controls, 15
 demographic information, 15
 Demographic Information, 15
 DIACRITICS, 16
 File management, 24
 Line comments, 74
 Lines and Rows in LIPP, 15
 loading files, 24
 MAIN SYMBOLS, 16
 Quit option, 14
 Save & Retrieve, 76
 Search & Replace, 76
 Selecting a file, 14
 Space information, 74
 tape counter numbers, 73
 Target Form Row, 15
 toggle features, 19
 Transcription Row, 15, 18
 word-processing conveniences, 15
 Transcription reliability, 42
 UPPER LIPP, 5
 word boundary, 23

Table of Contents – Appendices

[Appendix 1](#) - Keyboard Map

[Appendix 2](#) - 16 Dimensional Code for Symbols

[Appendix 3](#) - Titelipp symbol chart (meanings of symbols)

[Appendix 4](#) - Demonstration of transcription file

[Appendix 5](#) - Inventory analysis printout for Demo1.LIP

[Appendix 6](#) - Phonological process analysis rules
(Kcommon.LAL) with variable definitions

[Appendix 7](#) - Results file for analysis on Demo1.LIP using
Kcommon rules (Summary Only)

[Appendix 8](#) - Reliability analysis rules

[Appendix 9](#) - Demonstration reliability analysis file
(Relk&L.lip)

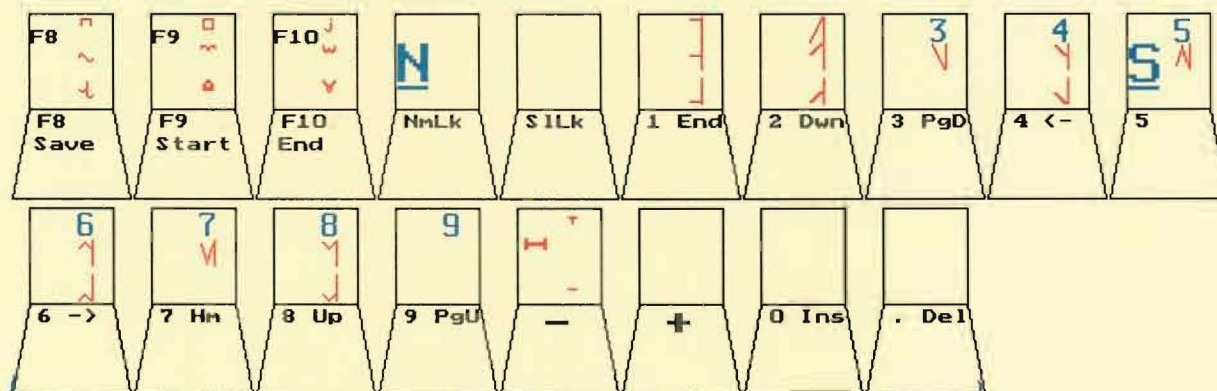
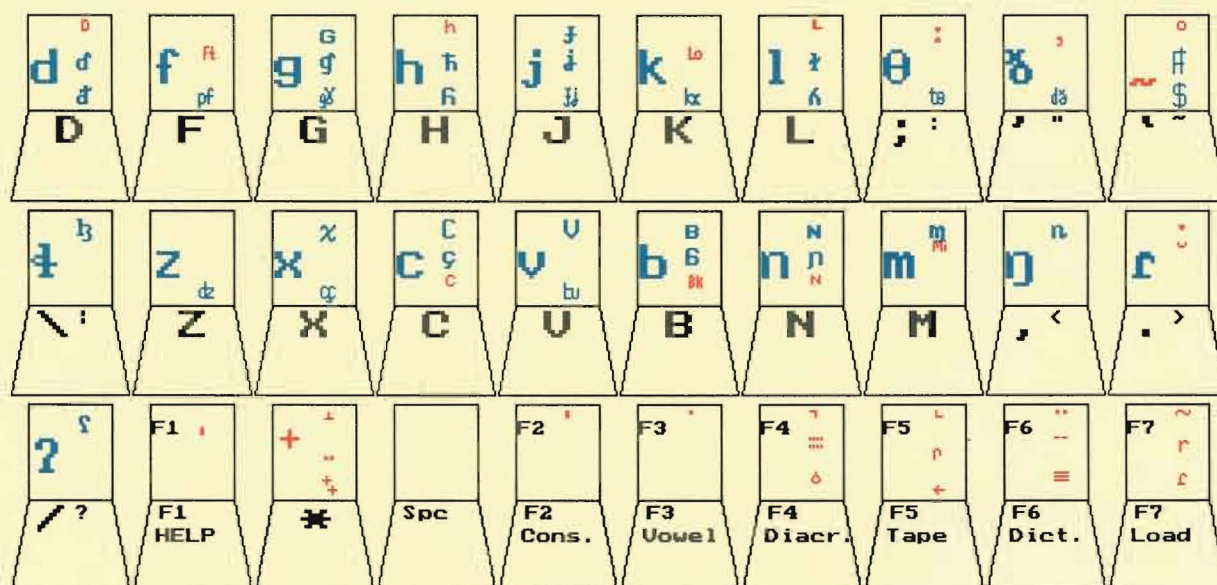
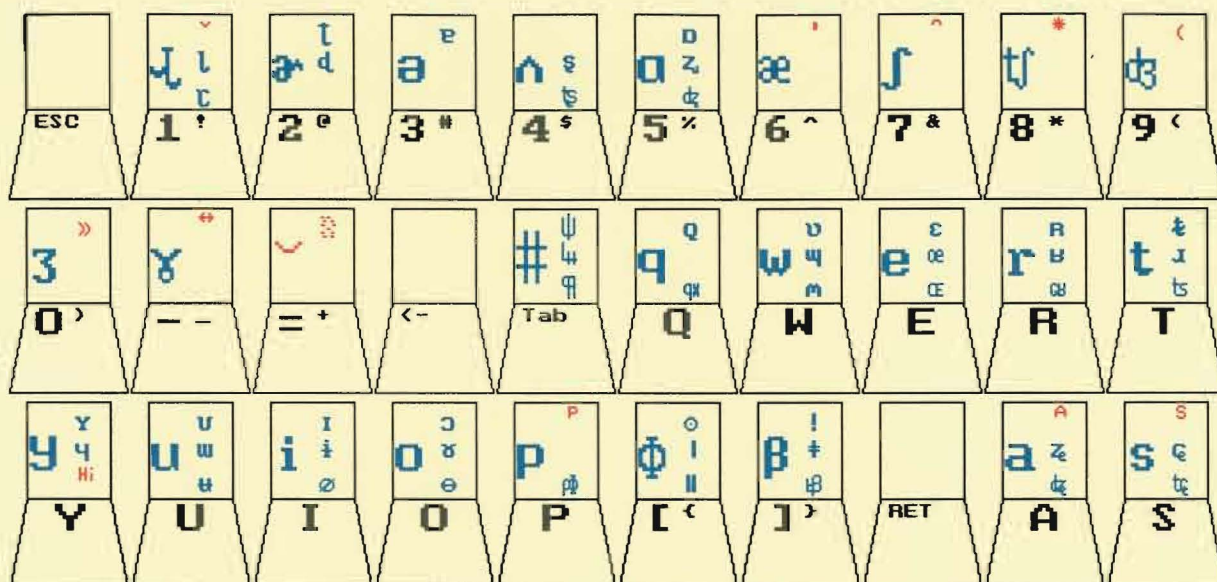
[Appendix 10](#) - The syntax of LAL (LIPP Analysis Language)

[Appendix 11](#) - Results file (all features) for MyFirst.LAL

[Appendix 12](#) - Demonstration infraphonological analysis and
report from (Canon.LAL)

Appendix 1 - Keyboard Map

(see following page)



Appendix 2 - 16 Dimensional Code for Symbols

<u>A</u> ClosureJaw	<u>B</u> TngFrntnes	<u>C</u> LipPosture	<u>D</u> VeloGlottFunc
[01]-Plosive [02]-Vibrant [03]-Fricative [04]-Affricate [05]-Trill [06]-Tap [07]-Nasal [08]-Semivowel [09]-UnspecC [11]-QuasiV [12]-HighV [13]-HighlaxV [14]-MidV [15]-MidlaxV [16]-LowV [17]-UnspecV	[01]-Dental [02]-Alveolar [03]-Palatoalv [04]-AlveoloPal [05]-Palatal [06]-Velar [07]-Uvular [08]-Pharyngeal [09]-Glottal [11]-FrontV [12]-FrntcentV [13]-CentralV [14]-BackcentV [15]-BackV	[01]-Bilabial [02]-LabioDental [03]-LabioLingual [04]-LabioPalatal [05]-LabioVelar [11]-HyperRound [12]-Round [13]-PartRound [14]-SpreadLips	[01]-Nasal [02]-SuctionClick [03]-Ejective [04]-Implosive
<u>E</u> GltSt&Voi	<u>F</u> OthMn,SecArt	<u>G</u> AnothOther	<u>H</u> YetAnother
[01]-UnvoiUnasp [02]-VoiUnasp [03]-UnvoiAsp [04]-VoiAsp [05]-Breathy [06]-Creaky [07]-VocTremor [08]-Falsetto	[01]-LabRound [02]-LbHfRdRCol [03]-LabFlat [04]-LabVelarized [05]-Velarized [06]-Palatalized [07]-ApproxAlv	[01]-Lateral [02]-ReleasedC [03]-UnreleasdC	[01]-Retroflex [02]-Bladed [03]-SlitArtic [04]-GroovedArt

<u>I</u> Strength	<u>J</u> Duration	<u>K</u> Tone	<u>L</u> ElemTypeSyl
[01]-Strong [02]-Weak [03]-Salivary [04]-Distorted [11]-Unstressed [12]-SecStressed [13]-PrimStressed	[01]-Long [02]-Short [03]-ExtrShort [11]-number3 [12]-number4 [13]-number5 [14]-number6 [15]-number7 [16]-number8 [17]-number9	[06]-HighLevel [07]-MidLevel [08]-LowLevel [09]-FullFall [10]-HighFall [11]-LowFall [12]-FullRise [13]-HighRise [14]-LowRise [15]-FullRiseFall [16]-HighRiseFall [17]-LowRiseFall [18]-FullFallRise [19]-HighFallRise [20]-LoFallRise	[01]-SlowTrnsMarg [02]-NonSylV [11]-SyllabicC

<u>M</u> UnspecGlobal	<u>N</u> MoreUnspecGlob	<u>O</u> (Not In Use)	<u>P</u> NonSpeech
[01]-UnspecC [02]-DorsalC [03]-CoronalC [04]-LabialC [09]-UnspecNuc [10]-UnspecSyl [11]-UnspecV [12]-HighV [13]-MidV [14]-LowV [15]-MoreMidV	[01]-ObstruentC [02]-LiquidC [03]-GlideC [11]-MoreCentV [12]-FrontV [13]-CentralV [14]-BackV [15]-BoundUTT [16]-BoundSYL [17]-BoundWORD [18]-BoundFOOT [19]-BoundLINE [20]-BoundPARAG		[10]-Unspec Nonsp [11]-Cry [12]-Laugh [13]-Sneeze [14]-Cough [15]-Hiccough [16]-Burp [17]-Grunt

Appendix 3 – Titelipp Symbol Definitions

		TITELIPP Character Definitions															
		Parameters															
Char:		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
	(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	(2)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	(3)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	(4)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	(5)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	(6)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
p	020	(7)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
b	021	(8)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t	022	(9)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
d	023	(10)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t	024	(11)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
d	025	(12)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c	026	(13)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f	027	(14)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
d	028	(15)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
k	030	(16)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g	031	(17)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	032	(18)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
q	033	(19)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10
G	034	(20)	1	0	1	0	1	0	0	0	0	0	0	4	1	0	0
2	035	(21)	1	0	1	0	2	0	0	0	0	0	0	4	1	0	0
	036	(22)	1	2	0	0	1	0	0	0	0	0	0	3	1	0	0
	037	(23)	1	2	0	0	2	0	0	0	0	0	0	3	1	0	0
	038	(24)	1	2	0	0	1	0	0	1	0	0	0	3	1	0	0
	039	(25)	1	2	0	0	2	0	0	1	0	0	0	3	1	0	0
	040	(26)	1	5	0	0	1	0	0	0	0	0	0	2	1	0	0
	041	(27)	1	5	0	0	2	0	0	0	0	0	0	2	1	0	0
	042	(28)	1	2	0	0	2	5	0	0	0	0	0	3	1	0	0
	043	(29)	1	2	0	0	1	5	0	0	0	0	0	3	1	0	0
	044	(30)	1	6	0	0	1	0	0	0	0	0	0	2	1	0	0
	045	(31)	1	6	0	0	2	0	0	0	0	0	0	2	1	0	0
	046	(32)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	047	(33)	1	7	0	0	1	0	0	0	0	0	0	2	1	0	0
	048	(34)	1	7	0	0	2	0	0	0	0	0	0	2	1	0	0
	049	(35)	1	9	0	0	1	0	0	0	0	0	0	1	0	0	0
	050	(36)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	051	(37)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	052	(38)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	053	(39)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	054	(40)	3	0	1	0	1	0	0	0	0	0	0	4	1	0	0
	055	(41)	3	0	1	0	2	0	0	0	0	0	0	4	1	0	0
	056	(42)	3	0	2	0	1	0	0	0	0	0	0	4	1	0	0
	057	(43)	3	0	2	0	2	0	0	0	0	0	0	4	1	0	0
	058	(44)	3	1	0	0	1	0	0	0	0	0	0	3	1	0	0
	059	(45)	3	1	0	0	2	0	0	0	0	0	0	3	1	0	0
	060	(46)	3	2	0	0	1	0	0	4	0	0	0	3	1	0	0
		(47)	3	2	0	0	2	0	0	4	0	0	0	3	1	0	0
		(48)	3	2	0	0	1	0	0	1	0	0	0	3	1	0	0
		(49)	3	2	0	0	2	0	1	0	0	0	0	3	1	0	0
		(50)	3	4	12	0	1	1	0	0	0	0	0	3	1	0	0
		(51)	3	4	12	0	2	1	0	0	0	0	0	3	1	0	0
		(52)	3	3	0	0	1	0	0	0	0	0	0	3	1	0	0
		(53)	3	4	0	0	2	0	0	0	0	0	0	3	1	0	0
		(54)	3	5	0	0	1	0	0	0	0	0	0	2	1	0	0
		(55)	3	5	0	0	2	0	0	0	0	0	0	2	1	0	0
		(56)	3	6	0	0	1	0	0	0	0	0	0	2	1	0	0
		(57)	3	6	0	0	2	0	0	0	0	0	0	2	1	0	0
		(58)	3	7	0	0	1	0	0	0	0	0	0	2	1	0	0
		(59)	3	7	0	0	2	0	0	0	0	0	0	2	1	0	0
		(60)	3	8	0	0	1	0	0	0	0	0	0	2	1	0	0

061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120

TITELIPP Character Definitions
Parameters

Char:	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
(61)	3	8	0	0	2	0	0	0	0	0	0	0	2	1	0	0
(62)	3	9	0	0	1	0	0	0	0	0	0	0	1	0	0	0
(63)	3	9	0	0	2	0	0	0	0	0	0	0	1	0	0	0
(64)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(65)	4	0	1	0	1	0	0	0	0	0	0	0	4	1	0	0
(66)	4	0	1	0	2	0	0	0	0	0	0	0	4	1	0	0
(67)	4	0	2	0	1	0	0	0	0	0	0	0	4	1	0	0
(68)	4	0	2	0	2	0	0	0	0	0	0	0	4	1	0	0
(69)	4	1	0	0	1	0	0	0	0	0	0	0	3	1	0	0
(70)	4	1	0	0	2	0	0	0	0	0	0	0	3	1	0	0
(71)	4	2	0	0	1	0	0	4	0	0	0	0	3	1	0	0
(72)	4	2	0	0	2	0	0	4	0	0	0	0	3	1	0	0
(73)	4	2	0	0	1	0	0	1	0	0	0	0	3	1	0	0
(74)	4	2	0	0	2	0	0	1	0	0	0	0	3	1	0	0
(75)	4	4	12	0	1	1	0	0	0	0	0	0	3	1	0	0
(76)	4	4	12	0	2	1	0	0	0	0	0	0	3	1	0	0
(77)	4	3	0	0	1	0	0	0	0	0	0	0	3	1	0	0
(78)	4	3	0	0	2	0	0	0	0	0	0	0	3	1	0	0
(79)	4	5	0	0	1	0	0	0	0	0	0	0	2	1	0	0
(80)	3	5	0	0	2	0	0	0	0	0	0	0	2	1	0	0
(81)	4	6	0	0	1	0	0	0	0	0	0	0	2	1	0	0
(82)	4	6	0	0	2	0	0	0	0	0	0	0	2	1	0	0
(83)	4	7	0	0	1	0	0	0	0	0	0	0	2	1	0	0
(84)	4	7	0	0	2	0	0	0	0	0	0	0	2	1	0	0
(85)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(86)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(87)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(88)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(89)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(90)	7	0	1	1	2	0	0	0	0	0	0	0	4	0	0	0
(91)	7	0	2	1	2	0	0	0	0	0	0	0	4	0	0	0
(92)	7	2	0	1	2	0	0	0	0	0	0	0	3	0	0	0
(93)	7	2	0	1	2	0	1	0	0	0	0	0	3	0	0	0
(94)	7	5	0	1	2	0	0	0	0	0	0	0	2	0	0	0
(95)	7	6	0	1	2	0	0	0	0	0	0	0	2	0	0	0
(96)	7	7	0	1	2	0	0	0	0	0	0	0	2	0	0	0
(97)	8	2	0	0	2	0	1	0	0	0	0	0	3	2	0	0
(98)	3	2	0	0	1	0	1	0	0	0	0	0	3	1	0	0
(99)	3	2	0	0	2	0	1	0	0	0	0	0	3	1	0	0
(100)	8	2	0	0	2	0	1	1	0	0	0	0	3	2	0	0
(101)	8	2	0	0	2	6	1	0	0	0	0	0	3	2	0	0
(102)	8	2	0	0	2	5	1	0	0	0	0	0	3	2	0	0
(103)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(104)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(105)	8	15	5	0	2	4	0	0	0	0	0	0	4	3	0	0
(106)	8	2	0	0	2	6	0	0	0	0	0	0	3	3	0	0
(107)	8	0	2	0	2	3	0	0	0	0	0	0	4	3	0	0
(108)	8	6	0	0	2	5	0	0	0	0	0	0	4	3	0	0
(109)	8	5	12	0	2	6	0	0	0	0	0	0	3	3	0	0
(110)	8	0	5	0	1	4	0	0	0	0	0	0	4	3	0	0
(111)	8	3	0	0	2	0	0	1	0	0	0	0	3	2	0	0
(112)	8	2	0	0	2	2	0	1	0	0	0	0	3	2	0	0
(113)	5	2	0	0	2	0	0	0	0	0	0	0	3	0	0	0
(114)	7	2	0	0	2	0	0	0	0	0	0	0	3	0	0	0
(115)	6	2	0	0	2	0	0	0	0	0	0	0	3	0	0	0
(116)	5	7	0	0	2	0	0	0	0	0	0	0	2	0	0	0
(117)	5	0	1	0	2	0	0	0	0	0	0	0	4	0	0	0
(118)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(119)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(120)	1	0	1	4	2	0	0	0	0	0	0	0	4	1	0	0

121																			
122																			
123																			
124	Char:	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P		
125	(121)	1	2	0	4	2	0	0	0	0	0	0	0	3	1	0	0		
126	(122)	1	6	0	4	2	0	0	0	0	0	0	0	2	1	0	0		
127	(123)	1	0	1	2	1	0	0	0	0	0	0	0	4	1	0	0		
128	(124)	1	2	0	2	1	0	0	0	0	0	0	0	3	1	0	0		
129	(125)	1	4	0	2	1	0	0	1	0	0	0	0	3	1	0	0		
130	(126)	1	2	0	2	1	0	1	0	0	0	0	0	3	1	0	0		
131	(127)	1	3	0	2	1	0	0	0	0	0	0	0	3	1	0	0		
132	(128)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
133	(129)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
134	(130)	12	11	0	0	2	6	0	0	0	0	0	0	12	12	0	0		
135	(131)	13	11	0	0	2	0	0	0	0	0	0	0	12	12	0	0		
136	(132)	14	11	0	0	2	0	0	0	0	0	0	0	13	12	0	0		
137	(133)	15	11	0	0	2	0	0	0	0	0	0	0	13	12	0	0		
138	(134)	16	11	0	0	2	0	0	0	0	0	0	0	14	12	0	0		
139	(135)	16	13	0	0	2	5	0	0	0	0	0	0	14	13	0	0		
140	(136)	14	13	0	0	2	0	0	0	0	0	0	0	13	13	0	0		
141	(137)	12	13	0	0	2	0	0	0	0	0	0	0	12	13	0	0		
142	(138)	12	15	0	0	2	4	0	0	0	0	0	0	12	14	0	0		
143	(139)	14	15	0	0	2	0	0	0	0	0	0	0	2	1	0	0		
144	(140)	15	15	0	0	2	0	0	0	0	0	0	0	13	14	0	0		
145	(141)	15	13	0	0	2	0	0	0	0	0	0	0	13	13	0	0		
146	(142)	16	15	0	0	2	0	0	0	0	0	0	0	14	14	0	0		
147	(143)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
148	(144)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
149	(145)	12	11	12	0	2	6	0	0	0	0	0	0	12	12	0	0		
150	(146)	13	11	12	0	2	0	0	0	0	0	0	0	12	12	0	0		
151	(147)	14	11	12	0	2	0	0	0	0	0	0	0	13	13	0	0		
152	(148)	15	11	12	0	2	0	0	0	0	0	0	0	13	12	0	0		
153	(149)	16	11	12	0	2	0	0	0	0	0	0	0	13	12	0	0		
154	(150)	14	13	12	0	2													

[illegible]

Appendix 4 - Demonstration of transcription file (Demo1.LIP).

(see following pages – [Apdx4.pdf](#))

Line: 1 " c r a y o n s "

k^hɹ̥ e j ã n z

k e a s

Comment:

Line: 2 " t h r e e "

θ ɹ i

t ɪ

Comment:

Line: 3 " b l a c k "

b l æ k

b a ʔ

Comment:

Line: 4 " r e d "

ɹ ɛː d

v ɛ

Comment:

Line: 5 " y e l l o w "

j ɛː l o w

j e j o

Comment:

Line: 6 " c h a i r "

tʃ^hɛ ɹ

t e

Comment:

Line: 7 " c u p s "

k^hʌ p s

k a ʔ

Comment:

Line: 8

" "

#

f ɔ ʔ

Line: 9

Comment:

" g u m "

k ã m

k ã

Line: 10

Comment:

" g l a s s e s "

k l æ s i z

k ' a s ' a

Line: 11

Comment:

" h a t "

h æ t

h a ʔ

Line: 12

Comment:

" l e a f "

l i f

j i

Line: 13

Comment:

" s h o e "

ʃ u

t v

Line: 14

Comment:

" s o a p "

s o w p

t ɔ ʔ

Comment:

Line: 15 " s p o o n "

s p ũ n

p v

Comment:

Line: 16 " s t r i n g "

s t ɹ ĩ ŋ

t ɪ

Comment:

Line: 17 " t e e t h "

t^h i θ

t ɪ

Comment:

Line: 18 " t h u m b "

θ ð m

t ʌ p

Comment:

Line: 19 " w a t c h "

w ɔ tʃ

v a t

Comment:

Line: 20 " z i p p e r "

z ' i p æ

t ' i t ' i

Comment:

Line: 21 " d o g g i e "

' d ɔ g i

' k ɔ g ' i

Comment:

Line: 22 " l a m b "

l æ m

n æ m

Comment:

Line: 23 " b a n a n a "

b æ n æ n æ

n æ n æ

Comment:

Line: 24 " h o r s e "

h ɔ s

o s

Comment:

Line: 25 " a p p l e "

æ p l

p æ b

Comment:

Line: 26 " r a t "

r æ t

æ t

Comment:

Line: 27 " c a t "

k æ t

t æ k

Comment:

Appendix 5a – Target Inventory analysis printout for Demo1.LIP

Inventory Analysis (Traget -> Transcription) File: a:demo1.inv

[illegible]

Appendix 5b – Transcription Inventory analysis printout for Demo1.LIP

Inventory Analysis (Transcription <- Target) File: a:demo1.inv

<-	↓ 0006	s 0003	n 0002	l 0002	w 0002	b 0001	d 0001	f 0001	θ 0001	z 0001	= 0025	
p <-	0001	p 0001	m 0001								= 0003	
b <-	p 0001	b 0001										= 0002
t <-	t 0002	θ 0002	tʃ 0002	p 0001	k 0001	s 0001	z 0001	ʃ 0001				= 0011
k <-	k 0004	t 0001	d 0001									= 0006
g <-	g 0001											= 0001
ʔ <-	p 0002	t 0002	0001	k 0001								= 0006
f <-	0001											= 0001
s <-	s 0002	z 0001										= 0003
h <-	h 0001											= 0001
m <-	m 0001											= 0001
n <-	n 0002	l 0001										= 0003
j <-	l 0002	j 0001										= 0003
v <-	w 0001	↓ 0001										= 0002
i <-	i 0002											= 0002
ɪ <-	i 0002	ɪ 0002	ɹ 0001									= 0005
e <-	e 0002	e 0001										= 0003
ɛ <-	ɛ 0001											= 0001
æ <-	æ 0001											= 0001
a <-	æ 0007	ə 0001	ɪ 0001	ʌ 0001	ɑ 0001							= 0011
ə <-	ə 0001											= 0001
ʌ <-	ʌ 0002											= 0002
u <-	u 0002											= 0002
o <-	l 0001	o 0001	ɔ 0001									= 0003
ɔ <-	0001	o 0001	ɔ 0001									= 0003
# <-	# 0054											= 0054

Appendix 6 - Phonological process analysis rules (Kcommon.LAL) with variable definitions

```
*****
*
* KimRule Lipp Analysis Rules - Written in Lipp Analysis Language (LAL)
* LAL by Rafael Delgado
* For use with LIPP Alphabets and Definitions
* Rules Developed by D. K. Oller
* Includes all variable definitions and globals as well as rules
*
*****

**** General categories ****

DEF VAR "C"      = { A[1-9] L[n11] };           % Consonant      %
DEF VAR "TC"     = { A[1-9] B[n9] N[n3] L[n11] }; % TrueConsonant %
DEF VAR "V"      = { A[11-17] L[n2] };          % Vowel         %
DEF VAR "TV"     = { A[12-16] L[n2] };          % True Vowel    %
DEF VAR "Nu1"    = { M[9-15] };                 % Nucleus1 (most) %
DEF VAR "Nu2"    = { A[11-17] };                 % Nucleus2 (less) %
DEF VAR "Nu3"    = { L[11] };                   % Nucleus3 (sylC) %
DEF VAR "GS"     = { I[12-13] };                % GenStressedSg %
DEF VAR "Un"     = { I[11] };                   % UnstressedSg  %
DEF VAR "Nil"    = { A[0] B[0] C[0] D[0] E[0] F[0] G[0]
                    H[0] I[0] J[0] K[0] L[0] M[0] N[0]
                    O[0] P[0] };                 % Blank Space   %
DEF VAR "#"      = { N[15-20] };                 % Boundary      %
DEF VAR "Any"    = { A[0-18] J[n13-19] N[n15-20] }; % AnySound or Nil %

***** Global general categories ****

DEF GLOB "Nu"    = ( Nu1 OR Nu2 OR Nu3 );        % Any Nucleus   %
DEF GLOB "Sg"    = ( C OR V );                  % Segment       %

***** Consonant Manner ****

DEF VAR "P"      = { A[1] };                     % Plosives      %
DEF VAR "S"      = { A[3] };                     % Spirant       %
DEF VAR "Fr"     = { A[2-4] };                   % Frication Sound %
DEF VAR "A"      = { A[4] };                     % Affricate     %
DEF VAR "N"      = { A[7] };                     % Nasal         %
DEF VAR "Se"     = { A[8] };                     % Semivowel Gen %
DEF VAR "Son"    = { A[5-8] };                   % Sonorant      %
DEF VAR "Tr"     = { A[5] };                     % Trill         %
DEF VAR "Tp"     = { A[6] };                     % Tap           %
DEF VAR "LS"     = { A[8] G[1] };                 % LateralSemiV %
DEF VAR "RS"     = { A[8] H[1] };                 % RetroflexSemiV %
DEF VAR "O1"     = { N[1] };                     % Obstruent 1   %
DEF VAR "O2"     = { A[1-4] };                   % Obstruent 2   %
DEF VAR "Vb"     = { A[2] };                     % Vibrant       %

***** Consonant Place ****

DEF VAR "La1"    = { C[1-5] };                   % Labial Cons 1 %
DEF VAR "La2"    = { M[4] };                     % Labial Cons 2 %
DEF VAR "La3"    = { C[1] };                     % Labial Cons 3 %
DEF VAR "Ld"     = { C[2] };                     % Labiodental   %
DEF VAR "De"     = { B[1] };                     % Dental Cons   %
DEF VAR "Al"     = { B[2] };                     % Alveolar Cons %
DEF VAR "Co1"    = { B[1-5] };                   % Coronal 1     %
DEF VAR "Co2"    = { M[3] };                     % Coronal 2     %
DEF VAR "Ve"     = { B[6] };                     % Velar Cons    %
DEF VAR "Do1"    = { B[6-8] };                   % Dorsal 1      %
DEF VAR "Do2"    = { M[2] };                     % Dorsal 2      %
DEF VAR "Gl"     = { B[9] };                     % Glottal Cons  %
```

***** Other Consonant Features *****

```

DEF VAR "Wk"      = { I[2] };           % Weak          %
DEF VAR "Vd"      = { E[2,4-8] };       % Voiced Cons   %
DEF VAR "Uv"      = { E[1,3] };         % Unvoiced Cons %
DEF VAR "Asp"     = { E[3-4] };         % Aspirated Cons %
DEF VAR "UnAsp"   = { E[1-2] };         % UnAspirated C %
DEF VAR "Rd1"     = { C[11-12] };       % Rounded 1     %
DEF VAR "Rd2"     = { F[1] };           % Rounded 2     %
DEF VAR "Lz"      = { F[1-4] };         % Labialized    %
DEF VAR "Pz"      = { F[6] };           % Palatalized   %
DEF VAR "Pa"      = { B[3-5] };         % Palatal       %
DEF VAR "Vz"      = { F[4-5] };         % Velarized     %
DEF VAR "Sy"      = { L[11] };         % Syllabic      %

```

***** Vowel Characteristics *****

```

DEF VAR "Hi"      = { A[12] };           % High Vowel    %
DEF VAR "GH1"     = { A[12-13] };        % Gen. High 1   %
DEF VAR "GH2"     = { M[12] };           % Gen. High 2   %
DEF VAR "Lo"      = { A[16] };           % Low           %
DEF VAR "GW1"     = { A[15-16] };        % Gen Low 1     %
DEF VAR "GW2"     = { M[14] };           % Gen Low 2     %
DEF VAR "Q"       = { A[11] };           % Quasivowel    %
DEF VAR "Mi"      = { A[14] };           % Mid Vowel     %
DEF VAR "GM1"     = { A[13-15] };        % Gen Mid 1     %
DEF VAR "GM2"     = { M[13] };           % Gen Mid 2     %
DEF VAR "MiLax"   = { A[15] };           %               %
DEF VAR "HiLax"   = { A[13] };           %               %
DEF VAR "Ft"      = { B[11] };           % Front Vowel   %
DEF VAR "GF1"     = { A[11-12] };        % Gen Front 1   %
DEF VAR "GF2"     = { N[12] };           % Gen Front 2   %
DEF VAR "Ce"      = { B[13] };           % Central       %
DEF VAR "Bk"      = { B[15] };           % Back Vowel    %
DEF VAR "GB1"     = { B[14-15] };        % Gen Back 1    %
DEF VAR "GB2"     = { N[14] };           % Gen Back 2    %
DEF VAR "NasV"    = { A[11-17] D[1] };   %               %
DEF VAR "NNV"     = { A[11-17] D[n1] };  % Nonnasal V    %

```

***** Global consonant features *****

```

DEF GLOB "AC"      = ( C AND NOT Gl );   % Articulated C %
DEF GLOB "Liq"     = ( LS OR RS);        % Liquid         %
DEF GLOB "Gld"     = ( Se AND (NOT Liq)); % Glide         %
DEF GLOB "O"       = ( O1 OR O2 );       % Obstruent      %
DEF GLOB "Son2"    = (Son AND (NOT Gld)); %Sonorants -glides%
DEF GLOB "Strid"   = ( Fr AND (NOT De)); % Strident      %
DEF GLOB "TpTr"    = ( Tp OR Tr );       % Tap or Trill   %
DEF GLOB "La"      = ( La1 OR La2 );     % Labial Cons    %
DEF GLOB "Co"      = ( Co1 OR Co2 );     % Coronal        %
DEF GLOB "Do"      = ( Do1 OR Do2 );     % Dorsal         %
DEF GLOB "Pal"     = ( Pz OR Pa );       % General Palatal %
DEF GLOB "FrtC"    = ((AC AND La) OR
                     (AC AND Co));       % FrontC        %
DEF GLOB "OrC"     = (AC AND NOT N);     % Oral Cons     %
DEF GLOB "AntC"    = (AC AND NOT Ve);    % Anterior C    %
DEF GLOB "NonN"    = (AC AND NOT N);     % NonNasal C    %
DEF GLOB "NonP"    = (AC AND NOT P);     % NonPlosive C  %
DEF GLOB "NonCo"   = (AC AND NOT Co);    % NonCoronal C  %
DEF GLOB "NonLa"   = (AC AND NOT La);    % NonLabial C   %
DEF GLOB "NonStrid" = (AC AND NOT Strid); % NonStrident C %
DEF GLOB "NonFr"   = (AC AND NOT Fr);    % NonFricative C %
DEF GLOB "NonPal"  = (AC AND NOT Pal);   % NonPalatal C  %
DEF GLOB "NonLiq"  = (AC AND NOT Liq);   % NonLiquid C   %
DEF GLOB "NonGld"  = (AC AND NOT Gld);   % NonGlide C    %
DEF GLOB "NonC"    = (Sg AND NOT AC);    % Non C         %

```

***** Global vowel features *****

```

DEF GLOB "UnV"      = ( Un and V);           %Unstressed vowel %
DEF GLOB "StgV"     = ( V and (not Un));      % Strong vowel %
DEF GLOB "UnC"      = ( Un and C);           %Unstressed cons %
DEF GLOB "StgC"     = ( C and (not Un));      % Strong cons %
DEF GLOB "GH"       = ( GH1 OR GH2 );        % General High %
DEF GLOB "GW"       = ( GW1 OR GW2 );        % General Low %
DEF GLOB "GM"       = ( GM1 OR GM2 );        % General Mid %
DEF GLOB "GF"       = ( GF1 OR GF2 );        % General Central %
DEF GLOB "GB"       = ( GB1 OR GB2 );        % General Back %
DEF GLOB "Rd"       = ( Rd1 OR Rd2 );        % Rounded %
DEF GLOB "NonRd"    = (Sg AND NOT Rd);       % NonRound C or V %
DEF GLOB "VorGld"   = ( V OR Gld );          % Vowels + Glides %

```

***** Sequences *****

```

DEF GLOB "AnyGp" = (( Any AnyGp )
                    OR Any );

DEF GLOB "NilGp" = ( ( Nil NilGp ) OR
                    Nil );           % Nil Group %
DEF GLOB "CGp"  = ( ( C CGp ) OR
                    C );             % Cons Group %
DEF GLOB "VGp"  = ( ( V Gld ) OR
                    V );             % V or Diphthong %

DEF GLOB "OrCGp" = ( ( OrC OrCGp ) OR
                    OrC );           % CGp All Oral %
DEF GLOB "NasGp" = ( ( N NasGp ) OR
                    ( OrCGp NasGp ) OR
                    N );             % CGp atleast1 N %

DEF GLOB "AntGp" = (( AntC AntGp ) OR
                    AntC );          % CGp all Anterior%
DEF GLOB "VeGp"  = (( Ve VeGp ) OR
                    ( AntGp VeGp ) OR
                    Ve );            % CGp atleast1 Ve %

DEF GLOB "NonLaGp" = (( NonLa NonLaGp ) OR
                    NonLa );         % CGp all -Labial %
DEF GLOB "LaGp"   = (( La LaGp ) OR
                    ( NonLaGp LaGp ) OR
                    La );            % CGp atleast1 La %

DEF GLOB "NonCoGp" = (( NonCo NonCoGp ) OR
                    NonCo );         % CGp all -Coronal%
DEF GLOB "CoGp"   = (( Co CoGp ) OR
                    ( NonCoGp CoGp ) OR
                    Co );            % CGp atleast1 Co %

DEF GLOB "Syl"    = ( Nu OR ( CGp Nu ) OR
                    ( Nu CGp ) OR
                    ( CGp Nu CGp )); % Syllable %

```

***** Expanded sequences, including nils *****

```

DEF GLOB "CGpX" = (( NilGp CGpX ) OR
                   ( CGp CGpX ) OR
                   ( CGp ) OR
                   ( CGp NilGp ) );   % CGp expanded w/ nils %

DEF GLOB "VGpX" = (( V Gld ) OR
                   ( V Nil ) OR
                   V );               % V and Diph expanded %

DEF GLOB "OrCGpX" = ( ( NilGp OrCGpX ) OR
                     ( OrCGp OrCGpX ) OR

```

```

( OrCGp NilGp ) OR
( OrCGp NilGp ) OR
( OrCGp NilGp ));      % OrCGp expanded      %

DEF GLOB "NasGpX" = ( ( NilGp NasGpX ) OR
( NasGp NasGpX ) OR
( NasGp NilGp ) OR
( NasGp NilGp ) OR
( NasGp NilGp ));      % NasGp expanded      %

DEF GLOB "AntGpX" = (( NilGp AntGpX ) OR
( AntGp AntGpX ) OR
( AntGp NilGp ) OR
( AntGp NilGp ) OR
( AntGp NilGp ));      % AntGp expanded      %

DEF GLOB "VeGpX" = (( NilGp VeGpX ) OR
( VeGp VeGpX ) OR
( VeGp NilGp ) OR
( VeGp NilGp ) OR
( VeGp NilGp ));      % VeGp expanded      %

DEF GLOB "NonLaGpX" = ( ( NilGp NonLaGpX ) OR
( NonLaGp NonLaGpX ) OR
( NonLaGp NilGp ) OR
( NonLaGp NilGp ) OR
( NonLaGp NilGp ));      % NonLaGp expanded      %

DEF GLOB "LaGpX" = ( ( NilGp LaGpX ) OR
( LaGp LaGpX ) OR
( LaGp NilGp ) OR
( LaGp NilGp ) OR
( LaGp NilGp ));      % LaGp expanded      %

DEF GLOB "NonCoGpX" = ( ( NilGp NonCoGpX ) OR
( NonCoGp NonCoGpX ) OR
( NonCoGp NilGp ) OR
( NonCoGp NilGp ) OR
( NonCoGp NilGp ));      % NonCoGp expanded      %

DEF GLOB "CoGpX" = ( ( NilGp CoGpX ) OR
( CoGp CoGpX ) OR
( CoGp NilGp ) OR
( CoGp NilGp ) OR
( CoGp NilGp ));      % CoGp expanded      %

DEF GLOB "AnyGpX" = (( (Sg or Sy) AnyGpX) OR
( NilGp AnyGpX) OR
( (Sg or Sy) NilGp ) OR
( (Sg or Sy) ));

*****

***** DELETIONS in order of damage *****

DEF EXEC RULE "StgSylDel" = (( StgV ) -> ( Nil OR C ));
DEF EXEC RULE "WeakSylDel" = ((( UnV ) -> ( nil )) or
(( UnC V ) -> (( UnC nil ) or ( nil nil ))) or
(( UnC C V ) ->
(( UnC C nil ) or ( UnC nil nil ) or
( nil nil nil ))));
DEF EXEC RULE "ClustDel" = ((( V OR # ) C CGp ( V OR # )) ->
(( V OR # ) NilGp ( V OR # )));
DEF EXEC RULE "InitCDel" = (( # C V ) -> ( # nil V ));
DEF EXEC RULE "MedCDel" = (( V NonGld V ) -> ( V Nil V ));
DEF EXEC RULE "ClustReduc" = (( C C ) -> (( C Nil ) OR ( Nil C )));
DEF EXEC RULE "FinCDel" = (( V NonGld # ) -> ( V Nil # ));
DEF EXEC RULE "DiphReduc" = (( V Gld ) -> ( V Nil ));
DEF EXEC RULE "GlotSubs" = (( AC ) -> ( Gl ));

*****

```

**** ADJUNCTIONS in order of damage ****

```
DEF EXEC RULE "StgSylAdj"      = (( Nil or C ) -> ( StgV ));
DEF EXEC RULE "WeakSylAdj"    = (( Nil or C ) -> ( UnV OR Q OR Sy ));
DEF EXEC RULE "ClustAdj"      = (( nil nil ) -> ( C C ));
DEF EXEC RULE "InitCADj"      = (( # nil V ) -> ( # C V ));
DEF EXEC RULE "ClustCreation" = ((( C Nil ) OR ( Nil C )) -> ( C C ));
DEF EXEC RULE "MedCADj"       = ((( V or Gld ) nil V ) ->
    (( V or Gld ) C V ));
DEF EXEC RULE "FinCADj"       = ((( V or Gld ) nil # ) ->
    (( V OR Gld ) NonGld # ));
DEF EXEC RULE "DiphCreation"  = (( V Nil ) -> ( V Gld ));
```

**** Summary of Deletions and Adjunctions *****

```
DEF EXEC RULE "StgSylError" = ( StgSylDel OR StgSylAdj );
DEF EXEC RULE "WeakSylError" = ( WeakSylDel OR WeakSylAdj );
DEF EXEC RULE "StgCError" = ( ClustDel OR ClustAdj OR InitCDel OR InitCADj
    OR InitCDel OR MedCDel OR MedCADj );
DEF EXEC RULE "WeakCError" = ( FinCDel OR FinCADj OR ClustReduc OR
    ClustCreation );
DEF EXEC RULE "TeenycError" = ( GlotSubs OR DiphReduc OR DiphCreation );
```

**** Consonant Substitutions: Pairs of common and uncommon processes ****

```
DEF EXEC RULE "Stopping"      = (( S OR A ) -> ( P ));
DEF EXEC RULE "Spirantization" = ( ( C and Not ( Fr )) -> ( Fr ));
DEF EXEC RULE "Gliding"       = (( Liq ) -> ( Gld ));
DEF EXEC RULE "Liquidizing"   = (( Gld ) -> ( Liq ));
DEF EXEC RULE "Deaspir"       = (( Asp ) -> ( C and not Asp ));
DEF EXEC RULE "Aspir"         = ( ( O and not Asp ) -> ( Asp ));
DEF EXEC RULE "FinDevoicing"  = ((( Vd AND O ) # ) -> ( Uv # ));
DEF EXEC RULE "FinVoicing"    = (( Uv # ) -> ( Vd # ));
DEF EXEC RULE "ConsFronting"  = ((( AC AND [B1:=B] ) ->
    ( AC AND [B1>B] )) OR
    (( Ld AND S ) -> ( La3 )));
DEF EXEC RULE "ConsBacking"   = ((( AC AND [B1:=B] ) ->
    ( AC AND [B1<B] )) OR ((La3 AND S) ->
    ( Ld ))) ;
DEF EXEC RULE "NasalStop"     = ( ( N ) -> ( OrC ));
DEF EXEC RULE "NasalCreation" = ( ( OrC ) -> ( N ));
DEF EXEC RULE "Vocalization"  = (( V Uv V ) -> ( V Vd V ));
DEF EXEC RULE "MedDeVoi"      = (( Sg Vd ( C OR V ) ) ->
    ( ( Sg OR NIL ) Uv ( C OR V )));
```

```
DEF EXEC RULE "CommonCSubs" = (Stopping OR Deaspir OR Gliding OR
    FinDevoicing OR ConsFronting
    OR Vocalization );
DEF EXEC RULE "UncommonCSubs" = ( Spirantization OR Liquidizing
    OR ConsBacking OR MedDeVoi OR Aspir
    OR FinVoicing );
```

**** Vowel Substitutions ****

```
DEF EXEC RULE "Levelling1"    = (( V AND Lo ) -> ( Q ));
DEF EXEC RULE "Levelling2"    = (( V AND NOT Lo ) -> ( Q ));
```

** teeny height errors **

```

DEF EXEC RULE "Raising1" = ((( HiLax ) -> ( Hi )) OR
                             (( MiLax ) -> ( Mi )) OR
                             (( Mi ) -> ( HiLax )) OR
                             (( Lo ) -> ( MiLax )));

DEF EXEC RULE "Lowering1" = ((( Hi ) -> ( HiLax )) OR
                              (( Mi ) -> ( MiLax )) OR
                              (( HiLax ) -> ( Mi )) OR
                              (( MiLax ) -> ( Lo )));

DEF EXEC RULE "TeenyHeight" = (Raising1 OR Lowering1);

*** small height errors ***

DEF RULE "Raising2" = ((( Mi ) -> ( Hi )) OR
                       (( MiLax ) -> ( HiLax )) OR
                       (( Lo ) -> ( Mi )));
DEF RULE "Lowering2" = ((( Hi ) -> ( Mi )) OR
                        (( HiLax ) -> ( MiLax )) OR
                        (( Mi ) -> ( Lo )));
DEF EXEC RULE "SmallHeight" = (Raising2 OR Lowering2);

*** big height errors ****

DEF RULE "Raising3" = ((( MiLax ) -> ( Hi )) OR
                       (( Lo ) -> ( HiLax )));
DEF RULE "Lowering3" = ((( Hi ) -> ( MiLax )) OR
                        (( HiLax ) -> ( Lo )));
DEF EXEC RULE "BigHeight" = (Raising3 OR Lowering3);

**** huge height errors ****

DEF EXEC RULE "Raising4" = (( Lo ) -> ( Hi ));
DEF EXEC RULE "Lowering4" = (( Hi ) -> ( Lo ));

DEF EXEC RULE "HugeHeight" = (Raising4 OR Lowering4);

**** small fronting errors ***

DEF RULE "Back1" = ((( Ft ) -> ( Ce )) OR
                    (( Ce ) -> ( Bk )));
DEF RULE "Forward1" = ((( Ce ) -> ( Ft )) OR
                       (( Bk ) -> ( Ce )));
DEF EXEC RULE "SmallFront" = ( Back1 OR Forward1 );

*** big fronting error ***

DEF EXEC RULE "Back2" = (( Ft ) -> ( Bk ));
DEF EXEC RULE "Forward2" = (( Bk ) -> ( Ft ));
DEF EXEC RULE "BigFront" = (Back2 OR Forward2);

*** other small vowel errors **

DEF EXEC RULE "SmallNasal" = ((( NasV ) -> ( NV )) OR
                              (( NV ) -> ( NasV )));
DEF EXEC RULE "SmallRd" = ((( V AND Rd ) -> ( V AND NOT Rd )) OR
                           (( V AND NOT Rd ) -> ( V AND Rd )));

DEF EXEC RULE "CommonVSubs" = ( SmallHeight OR TeenyHeight OR SmallNasal
                               OR SmallRd OR SmallFront );
DEF EXEC RULE "UncommonVSubs" = ( BigHeight OR HugeHeight OR Levelling1 OR
                                  Levelling2 OR BigFront );

*****

**** Simple Assimilation, Coalescence, and Metathesis ****
**** These assimilation rules involve no expanded sequences, and ****
**** consequently are applicable only across single syllables ****

```

**** with singleton consonants, and assimilations must be total,
 **** i.e., single feature assimilation is ignored. *****

```
DEF EXEC RULE "ProgConsAssim" = ((( C AND [B1:=A,B2:=B,B3:=C,
    B4:=G,B5:=H] ) V ( C AND NOT
    [B1=A,B2=B,B3=C,
    B4=G,B5=H] ) ) -> (( C AND
    [B1=A,B2=B,B3=C,
    B4=G,B5=H] ) V ( C AND
    [B1=A,B2=B,B3=C,
    B4=G,B5=H] )));

DEF EXEC RULE "RegConsAssim" = ((( C AND [B1:=A,B2:=B,B3:=C,
    B4:=G,B5:=H] ) V ( C AND ( NOT
    [B1=A,B2=B,B3=C,
    B4=G,B5=H] ) AND
    ( [B6:=A,B7:=B,B8:=C,
    B9:=G,B10:=H] ) ) ) -> (( C AND
    [B6=A,B7=B,B8=C,
    B9=G,B10=H] ) V ( C AND
    [B6=A,B7=B,B8=C,
    B9=G,B10=H] )));
```

```
DEF EXEC RULE "RegVowelAssim" = ((( V AND [B1:=A,B2:=B] ) ( C )
    ( V AND ( NOT [B1=A,B2=B] )
    AND ( [B3:=A,B4:=B] ) ) ) ->
    ( ( V AND [B3=A,B4=B] ) ( C )
    ( V AND [B3=A,B4=B] )));
```

```
DEF EXEC RULE "ProgVowelAssim" = ((( V AND [B1:=A,B2:=B] ) ( C )
    ( V AND ( NOT [B1=A,B2=B] )
    AND ( [B3:=A,B4:=B] ) ) ) ->
    ( ( V AND [B1=A,B2=B] ) ( C )
    ( V AND [B1=A,B2=B] )));
```

***** Coalescences *****

```
DEF ELRULE "Coales1" = (((Fr AND Co) P) -> ((Nil (Fr AND (NOT Co))) OR
    ((Fr AND (NOT Co)) Nil))) OR ((P (Fr AND Co)) ->
    ((Nil (Fr AND (NOT Co))) OR ((Fr AND (NOT Co)) Nil)));
DEF ELRULE "Coales2" = (((Fr (P and Co)) -> ((Nil (P AND (NOT Co))) OR
    ((P AND (NOT Co)) Nil))) OR ((P and Co) Fr) ->
    ((Nil (P AND (NOT Co))) OR ((P AND (NOT Co)) Nil)));
DEF ELRULE "Coales4" = (((Fr (P and Ve)) -> ((Nil (P AND (NOT Ve))) OR
    ((P AND (NOT Ve)) Nil))) OR ((P and Ve) Fr) ->
    ((Nil (P AND (NOT Ve))) OR ((P AND (NOT Ve)) Nil)));
DEF ELRULE "Coales3" = (((P and not Co) Liq) -> ((Nil (P AND Co)) OR
    ((P AND Co) Nil))) OR ((Liq (P and not Co)) ->
    ((Nil (P AND Co)) OR ((P AND Co) Nil)));
DEF ELRULE "Coales5" = (((P and not La) Liq) -> ((Nil (P AND La)) OR
    ((P AND La) Nil))) OR ((Liq (P and not La)) ->
    ((Nil (P AND La)) OR ((P AND La) Nil)));
DEF ELRULE "Coales6" = (((P and not Ve) Liq) -> ((Nil (P AND Ve)) OR
    ((P AND Ve) Nil))) OR ((Liq (P and not Ve)) ->
    ((Nil (P AND Ve)) OR ((P AND Ve) Nil)));
DEF ELRULE "Coales7" = (((P and not La) Liq) -> ((Nil (Fr AND La)) OR
    ((Fr AND La) Nil))) OR ((Liq (P and not La)) ->
    ((Nil (Fr AND La)) OR ((Fr AND La) Nil)));
DEF EXEC RULE "Coales" = (Coales1 OR Coales2 OR Coales3 OR Coales4 OR Coales5
    OR Coales6 OR Coales7);
```

*****Metathetic Rules*****

```
DEF ELRULE "Metath1" = (((C AND [B1:=A,B2:=B]) (Syl)
    (C AND (NOT [B1=A,B2=B]) AND ([B3:=A,B4:=B]))) ->
    ((C AND [B3=A,B4=B]) (Syl)
    (C AND [B1=A,B2=B])));
DEF ELRULE "Metath2" = (((C AND [B1:=A,B2:=B]) (Syl) (Syl)
    (C AND (NOT [B1=A,B2=B]) AND ([B3:=A,B4:=B]))) ->
```

```

((C AND [B3=A,B4=B]) (Syl) (Syl)
(C AND [B1=A,B2=B]));
DEF ELRULE "Metath3" = (((C AND [B1:=A,B2:=B]) (Syl) (Syl) (Syl)
(C AND (NOT [B1=A,B2=B]) AND ([B3:=A,B4:=B]))) ->
((C AND [B3=A,B4=B]) (Syl) (Syl) (Syl)
(C AND [B1=A,B2=B]));

DEF EXEC RULE "Metath" = ((Metath1 OR Metath2 OR Metath3));

```

**** THESE COUNTS TAKE STOCK OF EVERY ITEM IN THE SAMPLE. IF THE CHILD
 ***** REFUSES TO PRONOUNCE A PARTICULAR WORD, IT SHOULD BE ELIMINATED
 ***** FROM THE LIST BEFORE ANALYSIS, OR ELSE KRULE WILL TREAT THE WHOLE
 ***** WORD AS A DELETION.

```

DEF EXEC RULE "StgVCnt" = (( StgV ) -> ( Any ));
DEF EXEC RULE "WeakVCnt" = (( Sy OR UnV ) -> ( Any ));
DEF EXEC RULE "StgCCnt" = ((( ( # AC Nu ) OR
( # NilGp AC Nu ) OR
( # NilGp AC NilGp Nu ) OR
( # AC NilGp Nu ) ) -> ( # Any )) OR
((( Nu NonGld Nu ) OR
( Nu NilGp NonGld NilGp Nu ) OR
( Nu NonGld NilGp Nu ) OR
( Nu NilGp NonGld Nu ) ) -> ( Any )));

DEF EXEC RULE "FinCCnt" = ((( ( Nu OR Gld ) NonGld # ) OR
( Nu NilGp NonGld # ) OR
( Nu NilGp NonGld NilGp # ) OR
( Nu NonGld NilGp # ) ) -> ( AnyGp # ));

DEF EXEC RULE "Clus2" = (( ( # OR NonC ) NonGld NonGld (NonC or # ) )
-> ( ( # AnyGp ) OR AnyGp ));

DEF EXEC RULE "Clus3" = (( ( # OR NonC ) NonGld NonGld NonGld
(NonC OR # ) ) ->
( ( # AnyGp ) OR AnyGp ));

DEF EXEC RULE "Clus4" = (( ( # OR NonC ) C C C C (NonC OR # ) ) ->
( ( # Any ) OR Any ));

DEF EXEC RULE "Clus5" = (( ( # OR NonC ) C C C C C (NonC OR # ) ) ->
( ( # Any ) OR Any ));

DEF EXEC RULE "ClusCnt" = ( Clus2 OR Clus3 OR Clus4 OR Clus5 );

DEF CALC "WeakCCnt" = ( FinCCnt0 + ( 2 * Clus20 ) + ( 3 * Clus30 )
+ ( 4 * Clus40 ) + ( 5 * Clus50 ));

DEF EXEC RULE "ConsCnt" = (( C ) -> ( Any ));
DEF EXEC RULE "ACCnt" = (( AC ) -> ( Any ));
DEF EXEC RULE "TCCnt" = (( TC ) -> ( Any ));
DEF EXEC RULE "DiphCnt" = (( V Gld ) -> ( Any ));
DEF EXEC RULE "VowelCnt" = (( V ) -> ( Any ));

```

*** The value of PCCommon or "the Percent of processes that occur commonly
 *** in child speech" ranges from 0 to 1. It can be taken as a gross
 *** measure of the extent to which the subject utilizes processes
 *** (or if you wish "makes phonetic errors") consistent with
 *** commonly occurring patterns of early child speech (approx. two-
 *** yr.-old).

```

DISP CALC "NumberCommon" = (WeakSylDel0 + ClustReduc0 + FinCDe10
+ CommonCSubs0 + CommonVSubs0 );
DISP CALC "NumberUncommon" = (StgSylError0
+ WeakSylAdj0 + ClustCreation0 + InitCDe10
+ ClustDe10 + UnCommonCSubs0 + UncommonVSubs0 );
DISP CALC "PCCommon" = ( NumberCommon / ( NumberCommon + NumberUncommon ));

```

END RULES;

Appendix 7 - Results file for analysis on Demo1.LIP using Kcommon rules (Summary Only)

```

*****
Transcription File: DEMO1.LIP
=====
Calculation of :227 NUMBERCOMMON Addr:3405
-----
Value:      58.000
=====
Calculation of :228 NUMBERUNCOMMON Addr:3429
-----
Value:      9.000
=====
Calculation of :229 PCCOMMON Addr:3463
-----
Value:      0.866
=====
*****
GRAND TOTALS:
*****
      STGSYLDEL      Occurred:      0.00%      (0/32)
      WEAKSYLDEL      Occurred:     100.00%      (1/1)
      CLUSTDEL        Occurred:      0.00%      (0/10)
      INITCDEL        Occurred:     10.53%      (2/19)
      MEDCDEL         Occurred:      0.00%      (0/6)
      CLUSTREDUC       Occurred:     100.00%     (11/11)
      FINCDEL         Occurred:     53.33%      (8/15)
      DIPHREDUC       Occurred:     100.00%      (3/3)
      GLOTSUBS        Occurred:      8.06%      (5/62)
      STGSYLADJ        Occurred:      1.47%      (1/68)
      WEAKSYLADJ       Occurred:      0.00%      (0/68)
      CLUSTADJ         Occurred:      0.00%      (0/2)
      INITCADJ         Occurred:     100.00%      (1/1)
      CLUSTCREATION    Did Not Occur: 0.00%      (0/0)
      MEDCADJ          Did Not Occur: 0.00%      (0/0)
      FINCADJ          Did Not Occur: 0.00%      (0/0)
      DIPHCREATION     Did Not Occur: 0.00%      (0/0)
      STGSYLErrorR     Occurred:      1.00%      (1/100)
      WEAKSYLErrorR    Occurred:      1.45%      (1/69)
      STGCERROR        Occurred:      7.89%      (3/38)
      WEAKCERROR       Occurred:     73.08%     (19/26)
      TEENYCERROR      Occurred:     12.31%     (8/65)
      STOPPING         Occurred:     38.89%      (7/18)
      SPIRANTIZATION   Occurred:      0.00%      (0/46)
      GLIDING          Occurred:     23.08%      (3/13)
      LIQUIDING        Occurred:      0.00%      (0/5)
      DEASPIR          Occurred:     100.00%      (5/5)
      ASPIR            Occurred:      0.00%      (0/34)
      FINDEVOICING     Occurred:     33.33%      (1/3)
      FINVOICING       Occurred:      0.00%      (0/10)
      CONSFROTING     Occurred:      9.68%      (6/62)
      CONSBACKING      Occurred:      8.06%      (5/62)
      NASALSTOP        Occurred:     12.50%      (1/8)
      NASALCREATION    Occurred:      1.85%      (1/54)
      VOCALIZATION     Occurred:      0.00%      (0/2)
      MEDDEVOI         Occurred:      0.00%      (0/40)
      COMMONCSUBS      Occurred:     23.88%     (16/67)
      UNCOMMONCSUBS    Occurred:      6.94%      (5/72)
      LEVELLING1       Occurred:      0.00%      (0/9)
      LEVELLING2       Occurred:      0.00%      (0/24)
      RAISING1          Occurred:     15.38%      (4/26)
      LOWERING1         Occurred:     25.00%      (6/24)
      TEENYHEIGHT      Occurred:     30.30%     (10/33)
      SMALLHEIGHT       Occurred:      3.03%      (1/33)
      BIGHEIGHT         Occurred:      0.00%      (0/26)
      RAISING4          Occurred:      0.00%      (0/9)
      LOWERING4         Occurred:     14.29%      (1/7)
      HUGEHEIGHT        Occurred:      6.25%      (1/16)

```

SMALLFRONT	Occurred:	27.78%	(10/36)
BACK2	Occurred:	0.00%	(0/18)
FORWARD2	Occurred:	0.00%	(0/13)
BIGFRONT	Occurred:	0.00%	(0/31)
SMALLNASAL	Occurred:	12.12%	(4/33)
SMALLRD	Occurred:	3.03%	(1/33)
COMMONVSUBS	Occurred:	61.11%	(22/36)
UNCOMMONVSUBS	Occurred:	2.78%	(1/36)
PROGCONSASSIM	Occurred:	4.00%	(1/25)
REGCONSASSIM	Occurred:	4.00%	(1/25)
REGVOWELASSIM	Occurred:	0.00%	(0/7)
PROGVOWELASSIM	Occurred:	14.29%	(1/7)
COALES	Occurred:	0.00%	(0/8)
METATH	Occurred:	3.23%	(1/31)
STGVCNT	Occurred:	100.00%	(32/32)
WEAKVCNT	Occurred:	100.00%	(2/2)
STGCCNT	Occurred:	100.00%	(24/24)
FINCCNT	Occurred:	100.00%	(16/16)
CLUS2	Occurred:	100.00%	(8/8)
CLUS3	Occurred:	100.00%	(1/1)
CLUS4	Did Not Occur:	0.00%	(0/0)
CLUS5	Did Not Occur:	0.00%	(0/0)
CLUSCNT	Occurred:	100.00%	(9/9)
CONSCNT	Occurred:	100.00%	(64/64)
ACCNT	Occurred:	100.00%	(62/62)
TCCNT	Occurred:	100.00%	(57/57)
DIPHCNT	Occurred:	100.00%	(3/3)
VOWELCNT	Occurred:	100.00%	(33/33)

Appendix 8 - Reliability analysis rules

```
*****
* Reliability Lipp Analysis Rules - Written in Lipp Analysis Language (LAL)
* LAL by Rafael Delgado
* For use with LIPP Alphabets and Definitions
* Rules Developed by D. K. Oller
*****
```

Note: Definitions not printed, but required – see appendix 4 for definitions

```
***** DELETIONS *****
DEF RULE "StrgSylDel" = (( StgV ) -> ( Nil or C ));
DEF RULE "WeakSylDel" = (( UnV or Q or Sy ) -> ( nil or C ));
DEF RULE "ClusterDel" = ((( V OR # ) AC CGp ( V OR # ) ->
                        (( V OR # ) NilGp ( V OR # )));
DEF RULE "InitConsDel" = (( # AC V ) -> ( # nil V ));
DEF RULE "MedConsDel" = (( VorGld AC VorGld ) ->
                        ( VorGld Nil VorGld ));
DEF RULE "FinConsDel" = (( V NonGld # ) -> ( V Nil # ));
DEF RULE "ClusterReduc" = (( C C ) -> (( C Nil ) OR ( Nil C )));
DEF RULE "SylWeakning" = (( StgV ) -> ( UnV or Q or Sy ));
DEF RULE "DiphReduc" = (( V Gld ) -> ( V Nil ));
DEF RULE "NearConsDel" = (( AC ) -> ( Gl ));
```

```
***** ADJUNCTIONS *****
```

```
DEF RULE "StrgSylAdj" = (( C or Nil ) -> ( StgV ));
DEF RULE "WeakSylAdj" = (( C or nil ) -> ( UnV or Q or Sy ));
DEF RULE "ClusterAdj" = (( nil nil ) -> ( C C ));
DEF RULE "InitConsAdj" = (( # nil V ) -> ( # C V ));
DEF RULE "MedConsAdj" = (( VorGld nil VorGld ) ->
                        ( VorGld AC VorGld ));
DEF RULE "FinConsAdj" = ((( V or Gld ) nil # ) ->
                        (( V OR Gld ) C # ));
DEF RULE "ClusterCreation" = ((( C Nil ) OR ( Nil C )) -> ( C C ));
DEF RULE "SylStrgning" = (( UnV or Q or Sy ) -> ( StgV ));
DEF RULE "DiphCreation" = (( V Nil ) -> ( V Gld ));
DEF RULE "NearConsAdj" = (( Gl ) -> ( AC ));
```

```
*****Deletion and Adjunction Errors in order of damage *****
```

```
DEF EXEC RULE "StrgSyl" = (StrgSylDel OR StrgSylAdj);
DEF EXEC RULE "WeakSyl" = (WeakSylDel OR WeakSylAdj);
DEF EXEC RULE "ClusterFull" = (ClusterDel OR ClusterAdj);
DEF EXEC RULE "InitCons" = (InitConsDel OR InitConsAdj);
DEF EXEC RULE "MedCons" = (MedConsDel OR MedConsAdj);
DEF EXEC RULE "FinCons" = (FinConsDel OR FinConsAdj);
DEF EXEC RULE "ClusterPart" = (ClusterReduc OR ClusterCreation);
DEF EXEC RULE "SylStrength" = (SylStrgning OR SylWeakning );
DEF EXEC RULE "Diph" = (DiphCreation OR DiphReduc);
DEF EXEC RULE "NearCons" = (NearConsAdj OR NearConsDel);
```

```
***** Consonant Substitutions *****
```

```
** small manner errors **
```

```
DEF RULE "PFr" = ( ( P ) -> ( Fr ) );
DEF RULE "FrP" = ( ( Fr ) -> ( P ) );
DEF RULE "SA" = ( ( S ) -> ( A ) );
DEF RULE "AS" = ( ( A ) -> ( S ) );
DEF RULE "PSon" = ( ( P ) -> ( Son ) );
DEF RULE "SonP" = ( ( Son ) -> ( P ) );
```

```

DEF RULE "NSe"      = ( ( N ) -> ( Se ) );
DEF RULE "SeN"      = ( ( Se ) -> ( N ) );
DEF RULE "LiqGld"   = ( ( Liq ) -> ( Gld ) );
DEF RULE "GldLiq"   = ( ( Gld ) -> ( Liq ) );
DEF RULE "VbFr"     = ( ( Vb ) -> ( Fr ) );
DEF RULE "FrVb"     = ( ( Fr ) -> ( Vb ) );
DEF RULE "VbTr"     = ( ( Vb ) -> ( Tr ) );
DEF RULE "TrVb"     = ( ( Tr ) -> ( Vb ) );
DEF RULE "TptoTr"   = ( ( Tp ) -> ( Tr ) );
DEF RULE "TrTp"     = ( ( Tr ) -> ( Tp ) );
DEF RULE "TpTrSe"   = ( ( TpTr ) -> ( Se ) );
DEF RULE "SeTpTr"   = ( ( Se ) -> ( TpTr ) );

** big manner errors **

DEF RULE "PVb"      = ( ( P ) -> ( Vb ) );
DEF RULE "VbP"      = ( ( Vb ) -> ( P ) );
DEF RULE "VbTpSeN"  = ( ( Vb ) -> ( Tp OR Se OR N ) );
DEF RULE "TpSeNVb"  = ( ( Tp OR Se OR N ) -> ( Vb ) );
DEF RULE "FrSon"    = ( ( Fr ) -> ( Son ) );
DEF RULE "SonFr"    = ( ( Son ) -> ( Fr ) );
DEF RULE "NTpTr"    = ( ( N ) -> ( TpTr ) );
DEF RULE "TpTrN"    = ( ( TpTr ) -> ( N ) );

** executable manners **

DEF EXEC RULE "SmallManError" = ( PFr OR FrP OR AS OR SA OR PSON OR
    SonP OR NSe OR SeN OR LiqGld OR GldLiq OR
    VbFr OR FrVb OR VbTr OR TrVb OR
    TrTp OR TptoTr OR TpTrSe OR SeTpTr );
DEF EXEC RULE "BigManError"   = ( PVb OR VbP OR VbTpSeN OR TpSeNVb OR
    FrSon OR SonFr OR NTpTr OR TpTrN );

** teeny place errors ***

DEF RULE "LaLd"      = ( ( La3 ) -> ( Ld ) );
DEF RULE "LdLa"      = ( ( Ld ) -> ( La3 ) );
DEF RULE "DeAl"      = ( ( De ) -> ( Al ) );
DEF RULE "AlDe"      = ( ( Al ) -> ( De ) );
DEF RULE "DoPa"      = ( ( Do ) -> ( Pa ) );
DEF RULE "PaDo"      = ( ( Pa ) -> ( Do ) );
DEF EXEC RULE "TeenyPlace" = ( LaLd OR LdLa OR DeAl OR AlDe OR DoPa OR PaDo );

*** small place errors ****

DEF RULE "LaCo"      = ( ( La ) -> ( Co ) );
DEF RULE "CoLa"      = ( ( Co ) -> ( La ) );
DEF RULE "DoCo"      = ( ( Do ) -> ( Co ) );
DEF RULE "CoDo"      = ( ( Co ) -> ( Do ) );
DEF EXEC RULE "SmallPlace" = ( LaCo OR CoLa OR DoCo OR CoDo );

** big place errors **

DEF RULE "DoLa"      = ( ( Do ) -> ( La ) );
DEF RULE "LaDo"      = ( ( La ) -> ( Do ) );
DEF EXEC RULE "BigPlace" = ( DoLa OR LaDo );

** small voicing errors **

DEF RULE "Deasp"     = ( ( Asp ) -> ( UnAsp ) );
DEF RULE "Aspir"     = ( ( UnAsp ) -> ( Asp ) );
DEF RULE "DeVoi"     = ( ( Vd ) -> ( Uv ) );
DEF RULE "Voi"       = ( ( Uv ) -> ( Vd ) );
DEF RULE "Voicing"   = ( DeVoi OR Voi );
DEF RULE "Aspiration" = ( Deasp OR Aspir );
DEF EXEC RULE "SmallVoi" = ( Voicing OR Aspiration );

** big voicing error **

```

```

DEF EXEC RULE "BigVoi" = ((( Asp ) -> ( Vd )) OR
                           (( Vd ) -> ( Asp)));

*****

**** Vowel Substitutions ****

** teeny height errors **

DEF RULE "OneStepUp" = ((( HiLax ) -> ( Hi )) OR
                        (( MiLax ) -> ( Mi )) OR
                        (( Mi ) -> ( HiLax )) OR
                        (( Lo ) -> ( MiLax )));

DEF RULE "OneStepDown" = ((( Hi ) -> ( HiLax )) OR
                          (( Mi ) -> ( MiLax )) OR
                          (( HiLax ) -> ( Mi )) OR
                          (( MiLax ) -> ( Lo )));
DEF EXEC RULE "TeenyHeight" = (OneStepUp OR OneStepDown);

*** small height errors ***

DEF RULE "TwoStepsUp" = ((( Mi ) -> ( Hi )) OR
                        (( MiLax ) -> ( HiLax )) OR
                        (( Lo ) -> ( Mi )));
DEF RULE "TwoStepsDown" = ((( Hi ) -> ( Mi )) OR
                          (( HiLax ) -> ( MiLax )) OR
                          (( Mi ) -> ( Lo )));
DEF EXEC RULE "SmallHeight" = (TwoStepsUp OR TwoStepsDown);

*** big height errors ****

DEF RULE "ThreeStepsUp" = ((( MiLax ) -> ( Hi )) OR
                          (( Lo ) -> ( HiLax )));
DEF RULE "ThreeStepsDown" = ((( Hi ) -> ( MiLax )) OR
                             (( HiLax ) -> ( Lo )));
DEF EXEC RULE "BigHeight" = (ThreeStepsUp OR ThreeStepsDown);

**** huge height errors ****

DEF EXEC RULE "HugeHeight" = ((( Hi ) -> ( Lo )) OR
                              (( Lo ) -> ( Hi )));

**** small fronting errors ***

DEF RULE "OneStepBack" = ((( Ft ) -> ( Ce )) OR
                          (( Ce ) -> ( Bk )));
DEF RULE "OneStepForward" = ((( Ce ) -> ( Ft )) OR
                              (( Bk ) -> ( Ce )));
DEF EXEC RULE "SmallFront" = ( OneStepBack OR OneStepForward );

*** big fronting error ***

DEF EXEC RULE "BigFront" = ((( Ft ) -> ( Bk )) OR
                           (( Bk ) -> ( Ft )));

*** other small vowel errors **

DEF EXEC RULE "SmallNasal" = ((( NasV ) -> ( NNV )) OR
                              (( NNV ) -> ( NasV )));
DEF EXEC RULE "SmallRd" = ((( V AND Rd ) -> ( V AND NOT Rd )) OR
                           (( V AND NOT Rd ) -> ( V AND Rd )));

***** Counts *****

DEF EXEC RULE "StgVCnt" = (( StgV ) -> ( Any ));
DEF EXEC RULE "WeakVCnt" = (( Q OR Sy OR UnV ) -> ( Any ));
DEF EXEC RULE "StgCCnt" = ((( # AC Nu ) OR

```

```

( # NilGp AC Nu ) OR
( # NilGp AC NilGp Nu ) OR
( # AC NilGp Nu ) -> ( # Any ) OR
((( Nu NonGld Nu ) OR
( Nu NilGp NonGld NilGp Nu ) OR
( Nu NonGld NilGp Nu ) OR
( Nu NilGp NonGld Nu ) -> ( Any )));
DEF EXEC RULE "FinCCnt" = ((( ( Nu OR Gld ) NonGld # ) OR
( Nu NilGp NonGld # ) OR
( Nu NilGp NonGld NilGp # ) OR
( Nu NonGld NilGp # ) ) -> ( AnyGp # ));
DEF EXEC RULE "Clus2" = (( ( # OR NonC ) NonGld NonGld ( NonC or # ) )
-> ( ( # AnyGp ) OR AnyGp ));
DEF EXEC RULE "Clus3" = (( ( # OR NonC ) NonGld NonGld NonGld
( NonC OR # ) ) ->
( ( # AnyGp ) OR AnyGp ));
DEF EXEC RULE "Clus4" = (( ( # OR NonC ) C C C C ( NonC OR # ) ) ->
( ( # Any ) OR Any ));
DEF EXEC RULE "Clus5" = (( ( # OR NonC ) C C C C C ( NonC OR # ) ) ->
( ( # Any ) OR Any ));
DEF EXEC RULE "ClusCnt" = ( Clus2 OR Clus3 OR Clus4 OR Clus5 );
DEF EXEC RULE "ConsCnt" = (( C ) -> ( Any ));
DEF EXEC RULE "ACCnt" = (( AC ) -> ( Any ));
DEF EXEC RULE "TCCnt" = (( TC ) -> ( Any ));
DEF EXEC RULE "DiphCnt" = (( V Gld ) -> ( Any ));
DEF EXEC RULE "VowelCnt" = (( V ) -> ( Any ));
DEF EXEC RULE "GlCnt" = (( Gl ) -> ( Any ));

***** Calculations on reliability *****

DEF CALC "WeakCCnt" = ( FinCCnt0 + ( 2 * Clus20 ) + ( 3 * Clus30 )
+ ( 4 * Clus40 ) + ( 5 * Clus50 ));

DISP CALC "Denom" = ( StgVCnt0 + ( .5 * WeakVCnt0 ) + ( .5 * StgCCnt0 )
+ ( .25 * WeakCCnt0 ) + ( .125 * ( DiphCnt0 + GlCnt0 ) ) );

DISP CALC "Numer" = ( Denom - ( StrgSyl0 ) - ( .5 * WeakSyl0 )
- ( .5 * InitCons0 )
- ( .5 * MedCons0 ) - ( .25 * FinCons0 )
- ( .25 * ClusterPart0 ) - ( .5 * ClusterFull0 )
- ( .125 * Diph0 ) - ( .125 * NearCons0 )
- ( .125 * SylStrength0 ));
DISP CALC "StrucAgreemt" = ( Numer / Denom );
DISP CALC "ConsAgreemt" = (( ConsCnt0 - (.2 * SmallManError0 )
- (.4 * BigManError0 )
- (.1 * TeenyPlace0 )
- (.2 * SmallPlace0 )
- (.3 * BigPlace0 )
- (.15 * SmallVoi0 )
- (.3 * BigVoi0 ) ) / ConsCnt0 );

DISP CALC "VowAgreemt" = (( VowelCnt0 - (.1 * TeenyHeight0 )
- (.2 * SmallHeight0 )
- (.3 * BigHeight0 )
- (.4 * HugeHeight0 )
- (.2 * SmallFront0 )
- (.3 * BigFront0 )
- (.1 * SmallNasal0 )
- (.1 * SmallRd0 ) ) / VowelCnt0 );

DISP CALC "TotalAgreemt" = ( ((StrucAgreemt * 2) + VowAgreemt
+ ConsAgreemt) / 4 );

*****

END RULES;

```

Appendix 9 - Demonstration reliability analysis file (Relk&L.lip).

(see following pages) ([apdx9.pdf](#))

Line: 1

" "

k^ˈa t ɪ q

k^ˈa k i ə

Line: 2

Comment:

" c o o k i e "

k^hʊ t̃ ɪ̃

k^hʊ k^hi

Line: 3

Comment:

" b a l l " b a l l "

p^ho # p^ho:

p o # p o

Line: 4

Comment:

" t h e " b a l l "

t ɪ # p^hɔ

d ɪ # p ə

Line: 5

Comment:

" "

k^ha k

k a k

Line: 6

Comment:

" t h a t " b i r d i e "

t æ: # p v ɹ^ˈi

d æ # p^ˈu ɹ i

Line: 7

Comment:

" "

k a k

k^ha k

Comment:

Line: 8

" "

k i k ^ˈ a k

k^h i k ^ˈ ɔ k

Comment:

Line: 9

" t h a t " b i r d i e "

t a ^ɹ # p ^ˈ ʊ d ^ˈ i

d æ # p u r i

Comment:

Line: 10

" t h e " w a l l "

t ə # v ə

t^h U # w ɔ

Comment:

Line: 11

" n o "

n ə

h U

Comment:

Line: 12

" t h e " w a l l "

t ə # v ɔ

t^h U # w ɔ

Comment:

Line: 13

" t h a t " o h " g o o d "

t a ? # l ɔ o ^ˈ v # k v ^ˈ ɪ z -

d æ # u k w e ^ˈ q

Comment:

Line: 14

" g o o d "

t ^ˈ ə β ə # k v : t

d æ w u C ^ˈ ʊ

Comment:

Line: 15 " r i g h t " h e r e "

e j # h i ə

e h ' i ə

Comment:

Line: 16 " w a l l "

v ə

b w o

Comment:

Line: 17 " n o m n y "

' n - ō m i

' n ō m i

Comment:

Line: 18 " a " w a l l "

ə # v ɔ

h U w ' ə

Comment:

Line: 19 " r i g h t " h e r e "

v ə j # i

C q

Comment:

Line: 20 " r i g h t " h e r e "

v ə j # h i ə ~

w U h ' i ə

Comment:

Line: 21 " "

B : ɔ

B U

Comment:

Line: 22 " "

 # p v ε #

 # b w U #

Comment:

Line: 23 " c u p "

 # k^h_ɿ p #

 # k^ha p #

Comment:

Line: 24 " " c u p "

 # k^hi # k^ha p #

 # k^hi k^hɔ b #

Comment:

Appendix 10 - The syntax of LAL (LIPP Analysis Language)

LIPP Analysis Language (LAL) Description:

Possible Statement Types:

- a. DEF VAR XXX;
- b. DEF RULE XXX;
- c. DEF GLOB XXX;
- d. DEF ELRULE XXX;
- e. DEF EXEC RULE XXX;
- f. DEF EXEC ELRULE XXX;
- g. EXEC XXX;
- h. DEF CALC XXX;
- i. DISP CALC XXX;
- j. DISPLAY "TEXT TEXT", (XXX) ;
- k. CALCULATE (BufferM: = XXX) ;
- l. IF «XXX»<-->(YYY) THEN DISPLAY (XXX) ;
- m. END RULES;

where XXX and YYY represent various declarations allowed for each of the statement types, TEXT represents a string of characters, M represents a value from 0-9, and <--> represents =, >, < or combinations of the same. Each Statement may be up to a maximum of 5000 characters long (including spaces between variables etc.) or about 60 full screen lines.

1) Variable Declarations:

Syntax: DEF VAR XXX;

XXX has the following format:

"NAME" = { A..P[0..16, nO..16, nO..16] CHR[0..255]
DIA[0..255] }

where NAME is the name of the variable up to 15 characters long. Not all parameters A..P, CHR or DIA need be present. Variable declarations may occur anywhere in the program.

A range of values may also be specified by using a “-“ For example, to define characters 20 to 30 use CHR[20-30] in the definition.

2) Rule or Global Declarations:

Syntax: DEF RULE XXX;

DEF GLOB XXX;

XXX (maximum 15 characters) has the following possible formats:

A) "NAME" = (VARNAME1..VARNAME_n) ;

Indicates a sequence of symbols represented by variables VARNAME1..VARNAME_n. The variables must have been defined prior to their usage.

B) "NAME" = ((VARNAME_a) -> (VARNAME_b)) ;

Indicates a variable VARNAME_a in the target line becoming another variable VARNAME_b in the transcription line. Note that the both VARNAME_a and VARNAME_b must be surrounded by parenthesis and that the entire rule is also surrounded by parenthesis.

Variables may also be combined with in a rule using boolean logic (OR, AND and NOT) .For example VARNAME may also have the following formats:

- i) VARNAME_a OR VARNAME_b OR. .OR VARNAME_n
- ii) VARNAME_a AND VARNAME_b AND. .AND VARNAME_n
- iii) NOT VARNAME_a

All three symbols may also be combined in one statement:

iv) (VARNAME_a OR VARNAME_b) and (VARNAME_c OR VARNAME_d)

when combining ANDs and ORs, use parenthesis to determine the order in which the evaluations are to be made. If not separated by a parenthesis, AND and OR are evaluated from left to right in order of occurrence. For example the statement:

(VARNAME_a OR VARNAME_b AND VARNAME_c OR VARNAME_d AND NOT VARNAME_e)

will be evaluated in the following manner:

((VARNAME_a OR VARNAME_b) AND VARNAME_c)
VARNAME_e) OR VARNAME_d) AND (NOT

You may also use boolean logic to combine rules that have been previously declared.

STORING and COMPARING Parameter values to storage buffers:

In Addition to the use of variable names, you may also use buffers to store or compare parameter information about a particular space along a sequence. There are a total of 99 buffers labelled B1 through B99. To store a parameter value to a buffer use thefollowing syntax:

((VARNAME_a AND [B2:=P,...,B14:=A]) VARNAME_b)

In this example the value associated with parameter P of the position corresponding to VARNAMEa is STORED in buffer B2 and that of parameter A is stored in buffer B14. You may use as many assign statements as there are buffers. To compare a buffer to a parameter value simply replace the "储" (Storage) symbol with one of the following symbols:

= Equals to
< Less than
> Greater than
<= Less than
>= Greater than
<> Not equal to

Examples:

[B3=P,B4<>A,B7>=C,B8<D,B1<=D]

In addition to comparing a buffer to a parameter, you may also compare it to a particular value (0..99) .

Syntax: [B2>=12] will compare the value in B2 to 12,

if the value in B2 is greater or equal to 12, TRUE will be returned else FALSE will be returned.

Note that in both the compare and store statements, the buffer name (B1..899) must occur before the store (储) or compare symbols (=,>=,<=,<,> or <> followed by the parameter A..P or a constant 0..99.

You may use an AND or an OR to combine the store and compare to buffer statements to a particular position. The STORE statement will always return a true value for valuation, while the COMPARE statements will return a value depending on the result of the evaluation.

3) Exact Length Rule Declarations:

Syntax: DEF ELRULE "NAME" = XXXX;

where xxxx is the same as the rules and global declarations.

In Exact Length rules, the program is flagged to make sure the target form matches in length the transcription.

given: ((A) -> (8)), the number of segments used for A must equal the number of segments of B, and the A sequence must line up with the 8 sequence in the transcript.

This becomes important when using recursive rules, where the length of A and B is not explicitly stated in the rules.

4)Recursive Statements:

These definitions are used when it is necessary to define an analysis rule with itself. For example:

```
DEF RULE II AAAA " = ( XXXX AAAA );
```

or

```
DEF RULE " AAAA " = ( ( XXXX AAAA ) or ( XXXX ) );
```

Note that AAAA is used to define the rule itself. The term XXXX may represent any variable name or variable sequence as in the standard rules or global declarations. A recursive statement is always composed of two portions, a recursive portion and a rule termination portion. AAAA is the recursive portion of the above examples while xxxx is the rule termination portion. Under no circumstances may the recursive portion of the rule appear as the first statement of a rule or variable sequence. Since variable sequences are executed from left to right, having the recursive portion occur first, will cause the LAL analysis to be thrown into an endless loop.

For example:

```
DO NOT USE:  
DEF RULE " AAAA " = ( AAAA xxxx );
```

Recursive definitions may be used with almost any LAL statement. However, the implementation and interpretation of execution of these rules can become very complex, especially when using several interactive recursive statement, and as such, should only be used by experienced LAL users.

5) Execute Statement:

This is to tag a particular rule for evaluation by LIPP. Some rules may have been written as subsections of other major rules, but never intended for actual execution. For this reason, rules to be executed have an addition "EXEC" statement in their declaration.

Syntax:

```
DEF EXEC RULE "NAME" = XXXX;      To execute a standard rule  
DEF EXEC ELRULE "NAME" = XXXX; To execute an exact length  
rule.  
EXEC "NAME"; To execute a previously defined rule.
```

Note: Rules are executed in the order in which they were declared in the LAL file. Having multiple EXEC NAME; statements will not cause the program to execute a particular rule multiple times.

6) Metric Calculation statement:

This statement is used to define the calculation of a particular metric to be used in the interpretation of the analysis results.

Syntax:

```
DEF CALC "NAME" = XXXX;  
DISP CALC "NAME" =XXXX;
```

where XXXX is a calculation equation. For example:

(2*AAAA + 10.4*BBBB -CCCC/DDDD -123.1 + 8)

The terms AAAA, BBBB, CCCC and DDDD are one of the following:

a) The name of an executable rule followed by an C, O or P extension. For example, if the name of the rule is ConsDel, BBBB might be ConsDelO. The extension makes reference to whether **C**: the possibility of occurrence count or **O**: the occurrence count or **P**: the percent occurrence (occurrence count divided by possibility count multiplied by 100) is used in the metric calculations. These rules must have been defined before they can be used in any calculation statement.

b) The name of another metric calculation statement (DEF CALC or DISP CALC) . Note that real number constants may be used to multiply, divide, add or subtract values from the results of the analysis rules. In addition, the calculations are performed using the standard mathematical operation priorities, where divisions, followed by multiplications and then by additions and subtractions are conducted. Although parenthesis may not be required to define simple calculations, it is strongly recommended that they be used to subdivide very complex formulas for the user's benefit. The difference between the DEF and DISP syntax is that only the metrics defined with DISP statements are displayed in the results file. The DEF CALC definition was introduced in order to allow long calculations to be subdivided into modules that can be combined in other DEF CALC or DISP CALC statements.

CAUTION: Although LAL will allow the definition of a recursive metric calculation statement, this should not be done. There is no way to stop the recursion.

7) Display Statement:

This statement is used to create a line of text in a Report, with textual information enclosed in quotation marks.

Syntax:

Display " TEXT.....TEXT ,;

where TEXT can be any characters or blank spaces.

Display statements can also be used to specify the printing of information from rule outcomes or demographic information.

Syntax:

Display " TEXT.....TEXT ", (DEMO_XXXX) ;

Display" TEXT.....TEXT ", (YYYYZ) ;

where XXXX represents any of the following demographics terms:

NAME	= a string variable
ID	= a string variable, the "Local ID number"
AGE	= in days (numerical)
BDATE	= birth date
SDATE	= session date
GEST	= gestational age in weeks (numerical)
LANG	= a string variable, the "Languages"
DISA	= a string variable, the "Disability"
SEX	= M or F
COMM	= a string variable, the "Session Comment"
TRAN	= a string variable, the "Transcriber"
ALPH	= the alphabet name
FILE	= filename (DOS format)

YYYY represents the name of any execution rule or calculation rule in the analysis file, and Z (optional) represents O (Occurrence) , C (Count), or P (Percent) based on the outcome of an execution rule specified. For any variable in a Display Statement that represents a real number, the value can be truncated or rounded.

Syntax:

Display "TEXT.....TEXT ", Trunc(XXXX) ;

Display "TEXT.....TEXT ", Round (XXXX) ;

where XXXX represents any real number variable.

Display statements can also utilize arithmetic calculations to combine or modify numerical variable values. For example,

Display " ", (AAAA/m + (BBBB*n) –CCCC) ;

would display variable AAAA divided by m plus variable BBBB times n minus variable CCCC.

8) Report Generation Buffers:

There are ten buffers available to store information while complex calculations are being made for a report --Buffer0, Buffer1 ...Buffer9.

Syntax:

Calculate (BufferX: = YYYY) ;

where yyyy is any numerical variable including a previously set buffer value. Calculations can be conducted within buffer rules.

For example,

Calculate (BufferX: = BufferX + (BufferY * ZZZZ) ;

would add to the previous value of BufferX the value of BufferY times the numerical variable zzzz. Buffer values can be referred to as Display Statement variables.

9) IF/THEN Statements:

Display statements and calculations on buffer quantities can be made contingent on rule outcomes or calculation outcomes utilizing IF/THEN statements. For example:

IF ((SSSS) = (MMMM)) THEN
Display " ", (RRRR);

would display the value RRRR only if SSSS equals MMMM; the syntax allows >, <, =, >= (equal to or greater than), or <= (equal to or less than) relationships, and variables, such as SSS, MMM, can be buffer values, calculation rule outcomes or execution rule outcomes.

10) End Rules Statement:

An analysis file in LAL must end with an End Rules statement.

Syntax: End Rules;

Appendix 11 - Results file (all features) for MyFirst.LAL

Transcription File: DEMO1.LIP

Executing: Rule: 8 STOPPING Addr:44

possible: @ Line: 1 Pos: 8	
possible: @ Line: 2 Pos: 2	Rule Occurred!
possible: @ Line: 7 Pos: '5	
possible: @ Line: 10 Pos: 5	
possible: @ Line: 10 Pos: 7	
possible: @ Line: 11 Pos: 2	
possible: @ Line: 12 Pos: 4	
possible: @ Line: 13 Pos: 2	Rule Occurred!
possible: @ Line: 14 Pos: 2	Rule Occurred!
possible: @ Line: 15 Pos: 2	
possible: @ Line: 16 Pos: 2	
possible: @ Line: 17 Pos: 4	
possible: @ Line: 18 Pos: 2	Rule Occurred!
possible: @ Line: 20 Pos: 2	Rule Occurred!
possible: @ Line: 24 Pos: 2	
possible: @ Line: 24 Pos: 5	

Rule Occurred: 31.25% (5/16)

=====

Executing: Rule: 9 NASALIZATION Addr:49

possible: @ Line: 1 Pos: 2
possible: @ Line: 1 Pos: 3
possible: @ Line: 1 Pos: 5
possible: @ Line: 1 Pos: 8
possible: @ Line: 2 Pos: 2
possible: @ Line: 2 Pos: 3
possible: @ Line: 3 Pos: 2
possible: @ Line: 3 Pos: 3
possible: @ Line: 3 Pos: 5
possible: @ Line: 4 Pos: 2
possible: @ Line: 4 Pos: 4
possible: @ Line: 5 Pos: 2
possible: @ Line: 5 Pos: 4
possible: @ Line: 5 Pos: 7
possible: @ Line: 6 Pos: 2
possible: @ Line: 6 Pos: 4
possible: @ Line: 7 Pos: 2
possible: @ Line: 7 Pos: 4
possible: @ Line: 7 Pos: 5
possible: @ Line: 9 Pos: 2
possible: @ Line: 10 Pos: 2
possible: @ Line: 10 Pos: 3
possible: @ Line: 10 Pos: 5
possible: @ Line: 10 Pos: 7
possible: @ Line: 11 Pos: 2
possible: @ Line: 11 Pos: 4
possible: @ Line: 12 Pos: 2
possible: @ Line: 12 Pos: 4
possible: @ Line: 13 Pos: 2
possible: @ Line: 14 Pos: 2
possible: @ Line: 14 Pos: 4
possible: @ Line: 14 Pos: 5
possible: @ Line: 15 Pos: 2
possible: @ Line: 15 Pos: 3
possible: @ Line: 16 Pos: 2
possible: @ Line: 16 Pos: 3
possible: @ Line: 16 Pos: 4
possible: @ Line: 17 Pos: 2
possible: @ Line: 17 Pos: 4
possible: @ Line: 18 Pos: 2
possible: @ Line: 19 Pos: 2

possible:	@ Line:	19 Pos:	4	
possible:	@ Line:	20 Pos:	2	
possible:	@ Line:	20 Pos:	4	
possible:	@ Line:	21 Pos:	2	
possible:	@ Line:	21 Pos:	4	
possible:	@ Line:	22 Pos:	2	Rule Occurred!
possible:	@ Line:	23 Pos:	2	
possible:	@ Line:	24 Pos:	2	
possible:	@ Line:	24 Pos:	4	
possible:	@ Line:	24 Pos:	5	
possible:	@ Line:	25 Pos:	4	
possible:	@ Line:	25 Pos:	5	
possible:	@ Line:	26 Pos:	2	
possible:	@ Line:	26 Pos:	4	
possible:	@ Line:	27 Pos:	2	
possible:	@ Line:	27 Pos:	4	

Rule Occurred: 1.75% (1/57)

=====

GRAND TOTALS :

STOPPING	Occurred: 31.25%	(5/16)
NASALIZATION	Occurred: 1.75%	(1/57)

Appendix 12 - Demonstration infraphonological analysis and report from Canon.LAL

Appendix 12a – Canon.LAL Rule set:

```
*****
*   Infraphonological rules in LAL
*   LAL by Rafael Delgado
*   Rules by D. K. Oller
*   Canon.lal
*****

**** General categories ****

DEF VAR "C"      = { A[1-9] L[n11] };           % Consonant      %
DEF VAR "TC"     = { A[1-9] B[n9] N[n3] L[n11] }; % TrueConsonant %
DEF VAR "V"      = { A[12-17] L[n2,n1] };       % Vowel          %
DEF VAR "TV"     = { A[12-16] L[n2] };           % True Vowel     %
DEF VAR "Nu1"    = { M[9-15] };                 % Nucleus1 (most) %
DEF VAR "Nu2"    = { A[11-17] };                 % Nucleus2 (less) %
DEF VAR "Nu3"    = { L[11] };                   % Nucleus3 (sylC) %
DEF VAR "Q"      = { A[11] };                   % QuasiVowel     %
DEF VAR "GS"     = { I[12-13] };                 % GenStressedSg  %
DEF VAR "Un"     = { I[11] };                   % UnstressedSg   %
DEF VAR "Nil"    = { A[0] B[0] C[0] D[0] E[0] F[0] G[0]
                  H[0] I[0] J[0] K[0] L[0] M[0] N[0]
                  O[0] P[0] };                 % Blank Space    %
DEF VAR "#"      = { N[15-20] };                 % Boundary       %
DEF VAR "Any"    = { A[0-18] J[n13-19] N[n15-20] }; % AnySound or Nil %

***** Global general categories ****

DEF GLOB "Nu"    = ( Nu1 OR Nu2 OR Nu3 );         % Any Nucleus    %
DEF GLOB "Sg"    = ( C OR Nu );                  % Segment        %

***** Consonant Manner ****

DEF VAR "P"      = { A[1] };                     % Plosives       %
DEF VAR "S"      = { A[3] };                     % Spirant        %
DEF VAR "Fr"     = { A[2-4] };                   % Frication Sound %
DEF VAR "A"      = { A[4] };                     % Affricate      %
DEF VAR "N"      = { A[7] };                     % Nasal          %
DEF VAR "Se"     = { A[8] };                     % Semivowel Gen  %
DEF VAR "Son"    = { A[5-8] };                   % Sonorant       %
DEF VAR "Tr"     = { A[5] };                     % Trill          %
DEF VAR "Tp"     = { A[6] };                     % Tap            %
DEF VAR "LS"     = { A[8] G[1] };                 % LateralSemiV   %
DEF VAR "RS"     = { A[8] H[1] };                 % RetroflexSemiV %
DEF VAR "O1"     = { N[1] };                     % Obstruent 1    %
DEF VAR "O2"     = { A[1-4] };                   % Obstruent 2    %
DEF VAR "Vb"     = { A[2] };                     % Vibrant        %

***** Consonant Place ****

DEF VAR "La1"    = { C[1-5] };                   % Labial Cons 1  %
DEF VAR "La2"    = { M[4] };                     % Labial Cons 2  %
DEF VAR "La3"    = { C[1] };                     % Labial Cons 3  %
DEF VAR "Ld"     = { C[2] };                     % Labiodental    %
DEF VAR "De"     = { B[1] };                     % Dental Cons    %
DEF VAR "Al"     = { B[2] };                     % Alveolar Cons  %
DEF VAR "Co1"    = { B[1-5] };                   % Coronal 1      %
DEF VAR "Co2"    = { M[3] };                     % Coronal 2      %
DEF VAR "Ve"     = { B[6] };                     % Velar Cons     %
DEF VAR "Do1"    = { B[6-8] };                   % Dorsal 1       %
DEF VAR "Do2"    = { M[2] };                     % Dorsal 2       %
DEF VAR "Gl"     = { B[9] };                     % Glottal Cons   %
```

***** Other Consonant Features *****

```

DEF VAR "Wk"      = { I[2] };           % Weak          %
DEF VAR "Vd"      = { E[2,4-8] };       % Voiced Cons    %
DEF VAR "Uv"      = { E[1,3] };         % Unvoiced Cons  %
DEF VAR "Asp"     = { E[3-4] };         % Aspirated Cons %
DEF VAR "UnAsp"   = { E[1-2] };         % UnAspirated C  %
DEF VAR "Rd1"     = { C[11-12] };       % Rounded 1      %
DEF VAR "Rd2"     = { F[1] };           % Rounded 2      %
DEF VAR "Lz"      = { F[1-4] };         % Labialized     %
DEF VAR "Pz"      = { F[6] };           % Palatalized    %
DEF VAR "Pa"      = { B[3-5] };         % Palatal        %
DEF VAR "Vz"      = { F[4-5] };         % Velarized     %
DEF VAR "Sy"      = { L[11] };         % Syllabic       %

```

***** Vowel Characteristics *****

```

DEF VAR "Hi"      = { A[12] };           % High Vowel     %
DEF VAR "GH1"     = { A[12-13] };        % Gen. High 1    %
DEF VAR "GH2"     = { M[12] };           % Gen. High 2    %
DEF VAR "Lo"      = { A[16] };           % Low            %
DEF VAR "GW1"     = { A[15-16] };        % Gen Low 1      %
DEF VAR "GW2"     = { M[14] };           % Gen Low 2      %
DEF VAR "Mi"      = { A[14] };           % Mid Vowel      %
DEF VAR "GM1"     = { A[13-15] };        % Gen Mid 1      %
DEF VAR "GM2"     = { M[13] };           % Gen Mid 2      %
DEF VAR "MiLax"   = { A[15] };           %
DEF VAR "HiLax"   = { A[13] };           %
DEF VAR "Ft"      = { B[11] };           % Front Vowel    %
DEF VAR "GF1"     = { A[11-12] };        % Gen Front 1    %
DEF VAR "GF2"     = { N[12] };           % Gen Front 2    %
DEF VAR "Ce"      = { B[13] };           % Central         %
DEF VAR "Bk"      = { B[15] };           % Back Vowel     %
DEF VAR "GB1"     = { B[14-15] };        % Gen Back 1     %
DEF VAR "GB2"     = { N[14] };           % Gen Back 2     %
DEF VAR "NasV"    = { A[11-17] D[1] };   %
DEF VAR "NNV"     = { A[11-17] D[n1] };  % Nonnasal V     %

```

***** Global consonant features *****

```

DEF GLOB "AC"      = ( C AND NOT Gl );   % Articulated C  %
DEF GLOB "Liq"     = ( LS OR RS );       % Liquid          %
DEF GLOB "Gld"     = ( Se AND (NOT Liq)); % Glide          %
DEF GLOB "O"       = ( O1 OR O2 );       % Obstruent       %
DEF GLOB "Son2"    = ( Son AND (NOT Gld)); %Sonorants -glides%
DEF GLOB "Strid"   = ( Fr AND (NOT De)); % Strident        %
DEF GLOB "TpTr"    = ( Tp OR Tr );       % Tap or Trill    %
DEF GLOB "La"      = ( La1 OR La2 );     % Labial Cons     %
DEF GLOB "Co"      = ( Co1 OR Co2 );     % Coronal         %
DEF GLOB "Do"      = ( Do1 OR Do2 );     % Dorsal          %
DEF GLOB "Pal"     = ( Pz OR Pa );       % General Palatal %
DEF GLOB "FrtC"    = ((AC AND La) OR
                      (AC AND Co));       % FrontC          %
DEF GLOB "OrC"     = (AC AND NOT N);     % Oral Cons       %
DEF GLOB "AntC"    = (AC AND NOT Ve);    % Anterior C      %
DEF GLOB "NonN"    = (AC AND NOT N);     % NonNasal C      %
DEF GLOB "NonP"    = (AC AND NOT P);     % NonPlosive C    %
DEF GLOB "NonCo"   = (AC AND NOT Co);    % NonCoronal C    %
DEF GLOB "NonLa"   = (AC AND NOT La);    % NonLabial C     %
DEF GLOB "NonStrid" = (AC AND NOT Strid); % NonStrident C  %
DEF GLOB "NonFr"   = (AC AND NOT Fr);    % NonFricative C  %
DEF GLOB "NonPal"  = (AC AND NOT Pal);   % NonPalatal C    %
DEF GLOB "NonLiq"  = (AC AND NOT Liq);   % NonLiquid C     %
DEF GLOB "NonGld"  = (AC AND NOT Gld);   % NonGlide C      %

```

***** Global vowel features *****

```

DEF GLOB "UnV"     = ( Un and V );       %Unstressed vowel %

```

[illegible]

```

DEF GLOB "NasGpX" = ( ( NilGp NasGpX ) OR
                      ( NasGp NasGpX ) OR
                      ( NasGp NilGp ) OR
                      ( NasGp ) OR
                      ( NasGp NilGp )); % NasGp expanded %

DEF GLOB "AntGpX" = (( NilGp AntGpX ) OR
                      ( AntGp AntGpX ) OR
                      ( AntGp NilGp ) OR
                      ( AntGp ) OR
                      ( AntGp NilGp )); % AntGp expanded %

DEF GLOB "VeGpX" = (( NilGp VeGpX ) OR
                     ( VeGp VeGpX ) OR
                     ( VeGp NilGp ) OR
                     ( VeGp ) OR
                     ( VeGp NilGp )); % VeGp expanded %

DEF GLOB "NonLaGpX" = ( ( NilGp NonLaGpX ) OR
                        ( NonLaGp NonLaGpX ) OR
                        ( NonLaGp NilGp ) OR
                        ( NonLaGp ) OR
                        ( NonLaGp NilGp )); % NonLaGp expanded %

DEF GLOB "LaGpX" = ( ( NilGp LaGpX ) OR
                     ( LaGp LaGpX ) OR
                     ( LaGp NilGp ) OR
                     ( LaGp ) OR
                     ( LaGp NilGp )); % LaGp expanded %

DEF GLOB "NonCoGpX" = ( ( NilGp NonCoGpX ) OR
                        ( NonCoGp NonCoGpX ) OR
                        ( NonCoGp NilGp ) OR
                        ( NonCoGp ) OR
                        ( NonCoGp NilGp )); % NonCoGp expanded %

DEF GLOB "CoGpX" = ( ( NilGp CoGpX ) OR
                     ( CoGp CoGpX ) OR
                     ( CoGp NilGp ) OR
                     ( CoGp ) OR
                     ( CoGp NilGp )); % CoGp expanded %

DEF GLOB "AnyGpX" = (( (Sg or Sy) AnyGpX) OR
                      ( NilGp AnyGpX) OR
                      ( (Sg or Sy) NilGp ) OR
                      ( (Sg or Sy) ));

```

***** Infraphonological terms *****

```

DEF VAR "CanC" = { A[1-9] B[n9] L[n1,n11] };
DEF VAR "MarC" = { A[1-9] B[n9] L[1] };
DEF VAR "MarV" = { A[10-17] L[1] };
DEF GLOB "NotCanC" = ( # OR (Sg AND NOT CanC) );
DEF GLOB "NotMarC" = ( # OR (Sg AND NOT MarC) );
DEF GLOB "X" = ( # OR Sg OR NIL );
DEF GLOB "BON" = ( # OR NIL );

```

***** Infraphonological rules *****

```

DEF RULE "Can1" = (( X X X ) -> ( # V CanC ));
DEF RULE "Can2" = (( X X X ) -> ( Nil V CanC ));
DEF RULE "Can3" = (( X X ) -> ( CanC V ));
DEF RULE "Mar1" = (( X X X ) -> ( # V MarC ));
DEF RULE "Mar2" = (( X X X ) -> ( Nil V MarC ));
DEF RULE "Mar3" = (( X X ) -> ( MarC V ));
DEF RULE "Mar4" = (( X X X ) -> ( # MarV CanC ));

```

```

DEF RULE "Mar5" = (( X X X ) -> ( Nil MarV CanC ));
DEF RULE "Mar6" = (( X X X ) -> ( CanC MarV CanC ));

DEF EXEC RULE "CanSylCnt" = ( Can1 OR Can2 OR Can3 );
DEF EXEC RULE "MarSylCnt" = ( Mar1 OR Mar2 OR Mar3 OR Mar4 OR
                               Mar5 OR Mar6 );

DEF EXEC RULE "FullVCnt" = (( Any ) -> (V));
DEF EXEC RULE "QuasiVCnt" = (( Any ) -> (Q));
DEF EXEC RULE "SylCnt" = (( Any ) -> ( Nu ));
DEF EXEC RULE "UttCnt" = ((( X X X ) -> ( BON Sg BON ) )
                          OR (( X X X X ) -> ( BON Sg Sg BON ))
                          OR ((X X X X X ) -> ( BON Sg Sg Sg BON ))
                          OR ((X X X X X X ) -> ( BON Sg Sg Sg Sg BON )));

DISP CALC "CanBBRatio" = (CanSylCnt0 / UttCnt0);
DISP CALC "MarBBRatio" = (MarSylCnt0 / UttCnt0);
DISP CALC "FullVRatio" = (FullVCnt0 / UttCnt0);
DISP CALC "QuasiVRatio" = (QuasiVCnt0 / UttCnt0);

*****
**** Report on Canonical Babbling *****

DISPLAY " ";
DISPLAY " ";
DISPLAY " ";
DISPLAY " ";
DISPLAY " ";
DISPLAY " CANONICAL BABBLING ANALYSIS ";
DISPLAY " ";
DISPLAY " Name of Client : ",(DEMO_Name);
DISPLAY " Date of Birth : ",(DEMO_BDate);
DISPLAY " Date of Phonological : ",(DEMO_SDate);
DISPLAY " Age in Years : ",(DEMO_Age/365);
DISPLAY " Examiner's Name : ",(DEMO_Tran);
DISPLAY " Diagnostic Information : ",(DEMO_Disa);
DISPLAY " ";
DISPLAY " SUMMARY ";
DISPLAY " ";
DISPLAY " Number of Utterances : ",Trunc(UttCnt0);
DISPLAY " Number of Syllables : ",Trunc(SylCnt0);
DISPLAY " Number of QuasiVowels : ",Trunc(QuasiVCnt0);
DISPLAY " Number of Full Vowels : ",Trunc(FullVCnt0);
DISPLAY " Number of Canonical Syl's : ",Trunc(CanSylCnt0);
DISPLAY " Number of Marginal Syl's : ",Trunc(MarSylCnt0);
DISPLAY " ";
DISPLAY " Canonical Babbling Ratio : ",(CanBBRatio);
DISPLAY " Marginal Babbling Ratio : ",(MarBBRatio);
DISPLAY " Full Vowel Ratio : ",(FullVRatio);
DISPLAY " QuasiVowel Ratio : ",(QuasiVRatio);

*****

END RULES;

```

Appendix 12b - Canon.LAL Analysis Results:

Transcription File: BABY.LIP

=====

Executing: Rule:149 CANSYLCNT Addr:1378

Possible:	@ Line:	1 Pos:	1	
Possible:	@ Line:	1 Pos:	2	
Possible:	@ Line:	1 Pos:	3	
Possible:	@ Line:	1 Pos:	4	
Possible:	@ Line:	1 Pos:	5	
Possible:	@ Line:	1 Pos:	6	
Possible:	@ Line:	1 Pos:	7	
Possible:	@ Line:	1 Pos:	8	
Possible:	@ Line:	1 Pos:	9	
Possible:	@ Line:	1 Pos:	10	
Possible:	@ Line:	1 Pos:	11	
Possible:	@ Line:	1 Pos:	12	
Possible:	@ Line:	1 Pos:	13	
Possible:	@ Line:	1 Pos:	14	
Possible:	@ Line:	1 Pos:	15	
Possible:	@ Line:	1 Pos:	16	
Possible:	@ Line:	1 Pos:	17	
Possible:	@ Line:	1 Pos:	18	
Possible:	@ Line:	1 Pos:	19	
Possible:	@ Line:	1 Pos:	20	
Possible:	@ Line:	1 Pos:	21	
Possible:	@ Line:	1 Pos:	22	
Possible:	@ Line:	1 Pos:	23	
Possible:	@ Line:	1 Pos:	24	
Possible:	@ Line:	1 Pos:	25	- Occurred!
Possible:	@ Line:	1 Pos:	26	
Possible:	@ Line:	1 Pos:	27	- Occurred!
Possible:	@ Line:	1 Pos:	28	
Possible:	@ Line:	1 Pos:	29	
Possible:	@ Line:	1 Pos:	30	
Possible:	@ Line:	1 Pos:	31	
Possible:	@ Line:	1 Pos:	32	
Possible:	@ Line:	1 Pos:	33	
Possible:	@ Line:	2 Pos:	1	
Possible:	@ Line:	2 Pos:	2	- Occurred!
Possible:	@ Line:	2 Pos:	3	
Possible:	@ Line:	2 Pos:	4	
Possible:	@ Line:	2 Pos:	5	
Possible:	@ Line:	2 Pos:	6	
Possible:	@ Line:	2 Pos:	7	- Occurred!
Possible:	@ Line:	2 Pos:	8	
Possible:	@ Line:	2 Pos:	9	- Occurred!
Possible:	@ Line:	2 Pos:	10	
Possible:	@ Line:	2 Pos:	11	- Occurred!
Possible:	@ Line:	2 Pos:	12	
Possible:	@ Line:	2 Pos:	13	- Occurred!
Possible:	@ Line:	2 Pos:	14	
Possible:	@ Line:	2 Pos:	15	
Possible:	@ Line:	2 Pos:	16	
Possible:	@ Line:	2 Pos:	17	
Possible:	@ Line:	2 Pos:	18	
Possible:	@ Line:	2 Pos:	19	
Possible:	@ Line:	2 Pos:	20	
Possible:	@ Line:	2 Pos:	21	
Possible:	@ Line:	2 Pos:	22	
Possible:	@ Line:	2 Pos:	23	
Possible:	@ Line:	2 Pos:	24	
Possible:	@ Line:	2 Pos:	25	
Possible:	@ Line:	2 Pos:	26	
Possible:	@ Line:	2 Pos:	27	
Possible:	@ Line:	2 Pos:	28	- Occurred!
Possible:	@ Line:	2 Pos:	29	

```

-----
Occurred:      12.90%   (8/62)
=====
Executing: Rule:150 MARSYLCNT Addr:1386
-----
Possible: @ Line: 1 Pos: 1
Possible: @ Line: 1 Pos: 2
Possible: @ Line: 1 Pos: 3
Possible: @ Line: 1 Pos: 4
Possible: @ Line: 1 Pos: 5
Possible: @ Line: 1 Pos: 6
Possible: @ Line: 1 Pos: 7
Possible: @ Line: 1 Pos: 8
Possible: @ Line: 1 Pos: 9
Possible: @ Line: 1 Pos: 10
Possible: @ Line: 1 Pos: 11
Possible: @ Line: 1 Pos: 12
Possible: @ Line: 1 Pos: 13
Possible: @ Line: 1 Pos: 14
Possible: @ Line: 1 Pos: 15
Possible: @ Line: 1 Pos: 16
Possible: @ Line: 1 Pos: 17
Possible: @ Line: 1 Pos: 18
Possible: @ Line: 1 Pos: 19 -      Occurred!
Possible: @ Line: 1 Pos: 20
Possible: @ Line: 1 Pos: 21 -      Occurred!
Possible: @ Line: 1 Pos: 22
Possible: @ Line: 1 Pos: 23
Possible: @ Line: 1 Pos: 24
Possible: @ Line: 1 Pos: 25
Possible: @ Line: 1 Pos: 26
Possible: @ Line: 1 Pos: 27
Possible: @ Line: 1 Pos: 28
Possible: @ Line: 1 Pos: 29
Possible: @ Line: 1 Pos: 30
Possible: @ Line: 1 Pos: 31
Possible: @ Line: 1 Pos: 32 -      Occurred!
Possible: @ Line: 1 Pos: 33
Possible: @ Line: 2 Pos: 1
Possible: @ Line: 2 Pos: 2
Possible: @ Line: 2 Pos: 3
Possible: @ Line: 2 Pos: 4
Possible: @ Line: 2 Pos: 5 -      Occurred!
Possible: @ Line: 2 Pos: 6
Possible: @ Line: 2 Pos: 7
Possible: @ Line: 2 Pos: 8
Possible: @ Line: 2 Pos: 9
Possible: @ Line: 2 Pos: 10
Possible: @ Line: 2 Pos: 11
Possible: @ Line: 2 Pos: 12
Possible: @ Line: 2 Pos: 13
Possible: @ Line: 2 Pos: 14
Possible: @ Line: 2 Pos: 15
Possible: @ Line: 2 Pos: 16
Possible: @ Line: 2 Pos: 17
Possible: @ Line: 2 Pos: 18
Possible: @ Line: 2 Pos: 19
Possible: @ Line: 2 Pos: 20
Possible: @ Line: 2 Pos: 21
Possible: @ Line: 2 Pos: 22
Possible: @ Line: 2 Pos: 23 -      Occurred!
Possible: @ Line: 2 Pos: 24
Possible: @ Line: 2 Pos: 25
Possible: @ Line: 2 Pos: 26 -      Occurred!
Possible: @ Line: 2 Pos: 27
Possible: @ Line: 2 Pos: 28
Possible: @ Line: 2 Pos: 29
-----
Occurred:      9.68%   (6/62)
=====
Executing: Rule:151 FULLVCNT Addr:1403

```

```

-----
Possible: @ Line: 1 Pos: 2
Possible: @ Line: 1 Pos: 4
Possible: @ Line: 1 Pos: 6 - Occurred!
Possible: @ Line: 1 Pos: 8 - Occurred!
Possible: @ Line: 1 Pos: 10 - Occurred!
Possible: @ Line: 1 Pos: 11 - Occurred!
Possible: @ Line: 1 Pos: 13
Possible: @ Line: 1 Pos: 15
Possible: @ Line: 1 Pos: 17
Possible: @ Line: 1 Pos: 19
Possible: @ Line: 1 Pos: 20 - Occurred!
Possible: @ Line: 1 Pos: 22
Possible: @ Line: 1 Pos: 23
Possible: @ Line: 1 Pos: 25
Possible: @ Line: 1 Pos: 26 - Occurred!
Possible: @ Line: 1 Pos: 27
Possible: @ Line: 1 Pos: 28 - Occurred!
Possible: @ Line: 1 Pos: 30 - Occurred!
Possible: @ Line: 1 Pos: 32
Possible: @ Line: 1 Pos: 33 - Occurred!
Possible: @ Line: 2 Pos: 2
Possible: @ Line: 2 Pos: 3 - Occurred!
Possible: @ Line: 2 Pos: 5
Possible: @ Line: 2 Pos: 6 - Occurred!
Possible: @ Line: 2 Pos: 8 - Occurred!
Possible: @ Line: 2 Pos: 9
Possible: @ Line: 2 Pos: 10 - Occurred!
Possible: @ Line: 2 Pos: 12 - Occurred!
Possible: @ Line: 2 Pos: 13
Possible: @ Line: 2 Pos: 14 - Occurred!
Possible: @ Line: 2 Pos: 16
Possible: @ Line: 2 Pos: 18
Possible: @ Line: 2 Pos: 20
Possible: @ Line: 2 Pos: 21
Possible: @ Line: 2 Pos: 23
Possible: @ Line: 2 Pos: 24 - Occurred!
Possible: @ Line: 2 Pos: 26
Possible: @ Line: 2 Pos: 27 - Occurred!
Possible: @ Line: 2 Pos: 28
Possible: @ Line: 2 Pos: 29 - Occurred!
-----
Occurred: 45.00% (18/40)
=====
Executing: Rule:152 QUASIVCNT Addr:1408
-----
Possible: @ Line: 1 Pos: 2 - Occurred!
Possible: @ Line: 1 Pos: 4 - Occurred!
Possible: @ Line: 1 Pos: 6
Possible: @ Line: 1 Pos: 8
Possible: @ Line: 1 Pos: 10
Possible: @ Line: 1 Pos: 11
Possible: @ Line: 1 Pos: 13 - Occurred!
Possible: @ Line: 1 Pos: 15
Possible: @ Line: 1 Pos: 17
Possible: @ Line: 1 Pos: 19
Possible: @ Line: 1 Pos: 20
Possible: @ Line: 1 Pos: 22
Possible: @ Line: 1 Pos: 23
Possible: @ Line: 1 Pos: 25
Possible: @ Line: 1 Pos: 26
Possible: @ Line: 1 Pos: 27
Possible: @ Line: 1 Pos: 28
Possible: @ Line: 1 Pos: 30
Possible: @ Line: 1 Pos: 32
Possible: @ Line: 1 Pos: 33
Possible: @ Line: 2 Pos: 2
Possible: @ Line: 2 Pos: 3
Possible: @ Line: 2 Pos: 5
Possible: @ Line: 2 Pos: 6
Possible: @ Line: 2 Pos: 8

```

Possible:	@ Line:	2	Pos:	9	
Possible:	@ Line:	2	Pos:	10	
Possible:	@ Line:	2	Pos:	12	
Possible:	@ Line:	2	Pos:	13	
Possible:	@ Line:	2	Pos:	14	
Possible:	@ Line:	2	Pos:	16	-
Possible:	@ Line:	2	Pos:	18	-
Possible:	@ Line:	2	Pos:	20	-
Possible:	@ Line:	2	Pos:	21	-
Possible:	@ Line:	2	Pos:	23	
Possible:	@ Line:	2	Pos:	24	
Possible:	@ Line:	2	Pos:	26	
Possible:	@ Line:	2	Pos:	27	
Possible:	@ Line:	2	Pos:	28	
Possible:	@ Line:	2	Pos:	29	

Occurred: 17.50% (7/40)

=====

Executing: Rule:153 SYLCNT Addr:1413

Possible:	@ Line:	1	Pos:	2	-	Occurred!
Possible:	@ Line:	1	Pos:	4	-	Occurred!
Possible:	@ Line:	1	Pos:	6	-	Occurred!
Possible:	@ Line:	1	Pos:	8	-	Occurred!
Possible:	@ Line:	1	Pos:	10	-	Occurred!
Possible:	@ Line:	1	Pos:	11	-	Occurred!
Possible:	@ Line:	1	Pos:	13	-	Occurred!
Possible:	@ Line:	1	Pos:	15	-	Occurred!
Possible:	@ Line:	1	Pos:	17	-	Occurred!
Possible:	@ Line:	1	Pos:	19		
Possible:	@ Line:	1	Pos:	20	-	Occurred!
Possible:	@ Line:	1	Pos:	22	-	Occurred!
Possible:	@ Line:	1	Pos:	23		
Possible:	@ Line:	1	Pos:	25		
Possible:	@ Line:	1	Pos:	26	-	Occurred!
Possible:	@ Line:	1	Pos:	27		
Possible:	@ Line:	1	Pos:	28	-	Occurred!
Possible:	@ Line:	1	Pos:	30	-	Occurred!
Possible:	@ Line:	1	Pos:	32		
Possible:	@ Line:	1	Pos:	33	-	Occurred!
Possible:	@ Line:	2	Pos:	2		
Possible:	@ Line:	2	Pos:	3	-	Occurred!
Possible:	@ Line:	2	Pos:	5		
Possible:	@ Line:	2	Pos:	6	-	Occurred!
Possible:	@ Line:	2	Pos:	8	-	Occurred!
Possible:	@ Line:	2	Pos:	9		
Possible:	@ Line:	2	Pos:	10	-	Occurred!
Possible:	@ Line:	2	Pos:	12	-	Occurred!
Possible:	@ Line:	2	Pos:	13		
Possible:	@ Line:	2	Pos:	14	-	Occurred!
Possible:	@ Line:	2	Pos:	16	-	Occurred!
Possible:	@ Line:	2	Pos:	18	-	Occurred!
Possible:	@ Line:	2	Pos:	20	-	Occurred!
Possible:	@ Line:	2	Pos:	21	-	Occurred!
Possible:	@ Line:	2	Pos:	23		
Possible:	@ Line:	2	Pos:	24	-	Occurred!
Possible:	@ Line:	2	Pos:	26		
Possible:	@ Line:	2	Pos:	27	-	Occurred!
Possible:	@ Line:	2	Pos:	28		
Possible:	@ Line:	2	Pos:	29	-	Occurred!

Occurred: 70.00% (28/40)

=====

Executing: Rule:154 UTTCNT Addr:1418

Possible:	@ Line:	1	Pos:	1	-	Occurred!
Possible:	@ Line:	1	Pos:	2		
Possible:	@ Line:	1	Pos:	3	-	Occurred!
Possible:	@ Line:	1	Pos:	4		
Possible:	@ Line:	1	Pos:	5	-	Occurred!
Possible:	@ Line:	1	Pos:	6		

Possible:	@ Line:	1 Pos:	7	-	Occurred!
Possible:	@ Line:	1 Pos:	8		
Possible:	@ Line:	1 Pos:	9	-	Occurred!
Possible:	@ Line:	1 Pos:	10		
Possible:	@ Line:	1 Pos:	11		
Possible:	@ Line:	1 Pos:	12	-	Occurred!
Possible:	@ Line:	1 Pos:	13		
Possible:	@ Line:	1 Pos:	14	-	Occurred!
Possible:	@ Line:	1 Pos:	15		
Possible:	@ Line:	1 Pos:	16	-	Occurred!
Possible:	@ Line:	1 Pos:	17		
Possible:	@ Line:	1 Pos:	18	-	Occurred!
Possible:	@ Line:	1 Pos:	19		
Possible:	@ Line:	1 Pos:	20		
Possible:	@ Line:	1 Pos:	21	-	Occurred!
Possible:	@ Line:	1 Pos:	22		
Possible:	@ Line:	1 Pos:	23		
Possible:	@ Line:	1 Pos:	24	-	Occurred!
Possible:	@ Line:	1 Pos:	25		
Possible:	@ Line:	1 Pos:	26		
Possible:	@ Line:	1 Pos:	27		
Possible:	@ Line:	1 Pos:	28		
Possible:	@ Line:	1 Pos:	29	-	Occurred!
Possible:	@ Line:	1 Pos:	30		
Possible:	@ Line:	1 Pos:	31	-	Occurred!
Possible:	@ Line:	1 Pos:	32		
Possible:	@ Line:	2 Pos:	1	-	Occurred!
Possible:	@ Line:	2 Pos:	2		
Possible:	@ Line:	2 Pos:	3		
Possible:	@ Line:	2 Pos:	4	-	Occurred!
Possible:	@ Line:	2 Pos:	5		
Possible:	@ Line:	2 Pos:	6		
Possible:	@ Line:	2 Pos:	7	-	Occurred!
Possible:	@ Line:	2 Pos:	8		
Possible:	@ Line:	2 Pos:	9		
Possible:	@ Line:	2 Pos:	10		
Possible:	@ Line:	2 Pos:	11	-	Occurred!
Possible:	@ Line:	2 Pos:	12		
Possible:	@ Line:	2 Pos:	13		
Possible:	@ Line:	2 Pos:	14		
Possible:	@ Line:	2 Pos:	15	-	Occurred!
Possible:	@ Line:	2 Pos:	16		
Possible:	@ Line:	2 Pos:	17	-	Occurred!
Possible:	@ Line:	2 Pos:	18		
Possible:	@ Line:	2 Pos:	19	-	Occurred!
Possible:	@ Line:	2 Pos:	20		
Possible:	@ Line:	2 Pos:	21		
Possible:	@ Line:	2 Pos:	22	-	Occurred!
Possible:	@ Line:	2 Pos:	23		
Possible:	@ Line:	2 Pos:	24		
Possible:	@ Line:	2 Pos:	25	-	Occurred!
Possible:	@ Line:	2 Pos:	26		
Possible:	@ Line:	2 Pos:	27		
Possible:	@ Line:	2 Pos:	28		

Occurred: 36.67% (22/60)

=====

Calculation of :155 CANBBRATIO Addr:1493

Value: 0.364

=====

Calculation of :156 MARBBRATIO Addr:1502

Value: 0.273

=====

Calculation of :157 FULLVRATIO Addr:1511

Value: 0.818

=====

Calculation of :158 QUASIVRATIO Addr:1520

Value: 0.318

=====

GRAND TOTALS:

CANSYLCNT	Occurred:	12.90%	(8/62)
MARSYLCNT	Occurred:	9.68%	(6/62)
FULLVCNT	Occurred:	45.00%	(18/40)
QUASIVCNT	Occurred:	17.50%	(7/40)
SYLCNT	Occurred:	70.00%	(28/40)
UTCNT	Occurred:	36.67%	(22/60)

CANONICAL BABBLING ANALYSIS

Name of Client : Isadora Frump
Date of Birth : 12/25/1990
Date of Phonological : 6/6/1991
Age in Years : 0.447
Examiners Name : Oller
Diagnostic Information : Very young

SUMMARY

Number of Utterances	:	22
Number of Syllables	:	28
Number of QuasiVowels	:	7
Number of Full Vowels	:	18
Number of Canonical Syls	:	8
Number of Marginal Syls	:	6
Canonical Babbling Ratio	:	0.364
Marginal Babbling Ratio	:	0.273
Full Vowel Ratio	:	0.818
QuasiVowel Ratio	:	0.318