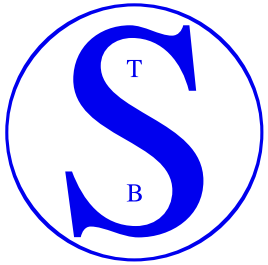
Conversion of legacy linguistic transcription data to *Unicode 4.0*



Richard S. Cook
Linguistics Department
STEDT Project
University of California, Berkeley
<rscook@socrates.berkeley.edu>
<http://stedt.berkeley.edu/>

Linguistic Society of America (LSA) 2005, Oakland, CA

Title: **"Conversion of legacy linguistic transcription data to Unicode 4.0"**

Speaker: Dr. Richard S. Cook

Workshop: Unicode for Linguists: How To Type, Send, and Archive Linguistic Texts by Computer
Time: Sunday, 9 January 9:00 AM - 12:00 PM
Place: Simmons 3/4, Oakland Convention Center
Slot: 10:30-11:00
Organizer: Deborah Anderson (UC-Berkeley)
Participants: Charles A. Bigelow (Bigelow & Holmes), William Bright, Peter Constable (Microsoft), Richard Cook (UC Berkeley), Kenneth Whistler (Sybase, Inc.)

CONVERSION OF LEGACY LINGUISTIC TRANSCRIPTION DATA TO UNICODE 4.0

Richard Cook <rscook@unicode.org>

LSA Workshop 2005/01/09

Re-representing (in Unicode) a legacy (non-Unicode) representation

TERMINOLOGY:

LEGACY DATA, CODE POINT (CODEPOINT), PLAIN TEXT, ENCODING, CUSTOM ENCODING, STANDARD ENCODING, ENCODING MODEL, UNICODE SCALAR VALUE, USV, MAPPING, CHARACTER SET (CHARACTERSET), ENCODING FORM, NORMALIZATION

LEGACY DATA

Most broadly: anything written, or a representation of anything written, from inscriptional or epigraphic material, to clay or silk or paper books, to graphical images, to pre-Unicode 4.X computer data.

Strictly: pre-Unicode 4.X computer data.

CODE POINT (CODEPOINT, CODE POSITION)

Any integer numeric value in the range 0 to 10FFFF_16 (i.e. 0 to 1,114,111) in the Unicode codespace.
See The Unicode Standard 4.0, "D4b, Section 3.4, Characters and Encoding"

ENCODING

A system for associating specific symbols each with specific numbers. The "abstract character" together with its associated USV (code point) constitutes an "Encoded Character".

CUSTOM ENCODING

A relatively non-standard encoding.

STANDARD ENCODING

A wide-spread encoding system, in general or universal use among users in a particular community.

ENCODING MODEL

The design principles behind a given encoding. This includes, for example, such things as the criteria for

distinguishing encoded entities, and the ordering of base character and diacritic.

MAPPING (MAPPING TABLE)

(1) Association of abstract character and code point.
(2) Association between code points (within a single encoding, or between different encodings).

CHARACTER SET (CHARACTERSET)

A collection of text elements, often within a particular encoding.

ENCODING FORM (CHARACTER ENCODING FORM)

Mapping from a character set definition to the actual CODE UNITS used to represent the data.

CODE UNIT

The minimal bit combination that can represent a unit of encoded text. The Unicode Standard uses 8-bit code units in the UTF-8 encoding form, 16-bit code units in the UTF-16 encoding form, and 32-bit code units in the UTF-32

encoding form. (See definition D28a in Section 3.9, Unicode Encoding Forms.)

STRING

Sequence of code points in computerized text. Non-Unicode-aware software may confuse code point with code unit.

PLAIN TEXT

Computer-encoded text that consists *only* of a sequence of code points from a given encoding standard, with no other formatting or structural information. Plain text interchange is commonly used between computer systems that do not share higher-level protocols.

RICH TEXT (STYLED TEXT, FANCY TEXT, FORMATTED TEXT)

The opposite of plain text. The result of adding information to plain text. Applying specific font face, color, formatting information, etc. to plain text, one gets rich text. Converting rich text to plain text, information may be lost.

UNICODE SCALAR VALUE (USV)

Any Unicode code point except high-surrogate and low-surrogate code points. As a result of this definition, the set of Unicode scalar values consists of the ranges 0 to D7FF_16 and E000_16 to 10FFFF_16, inclusive. From The Unicode Standard 4.0, "D28, Section 3.9, Unicode Encoding Forms" <http://www.unicode.org/versions/Unicode4.0.0/ch03.pdf>

USV

A "U+" prefixed upper- or lower-case hexadecimal representation of *any* UCS code point (not necessarily excluding surrogates). Matches the regex /^U\+[0-9A-Fa-f]{4,6}$/ With bracketing, e.g. [U+7329][U+9be8], a fall-back encoding form (in the WenLin text editor).

SCALAR

1. <mathematics> A single number, as opposed to a vector or matrix of numbers. Thus, for example, "scalar multiplication" refers to the operation of multiplying one number (one scalar) by another and is used to contrast this with "matrix multiplication" etc.

2. <architecture> In a parallel processor or vector processor, the "scalar processor" handles all the sequential operations - those which cannot be parallelised or vectorised. See also superscalar (uniprocessor).

3. <programming> Any data type that stores a single value (e.g. a number or Boolean), as opposed to an aggregate data type that has many elements. A string is regarded as a scalar in some languages (e.g. Perl) and a vector of characters in others (e.g. {C}).
From <http://dictionary.reference.com/search?q=scalar>.

NORMALIZATION

A process of removing alternate representations of equivalent sequences from textual data, to convert the data into a form that can be binary-compared for equivalence.

CONVERSION PROCESS:

(0) Realizing that it needed to be done, learning how to do it, and that it could only be done in stages;

(1) Mapping to Unicode 3.0;

(2) Formal proposal of unencoded characters for inclusion in Unicode/ISO 10646;

(3) Mapping to Unicode 4.0;

(4) Creation of tools for conversion of the custom-encoded data;

(5) Creation of tools for use of the new Unicode data.

Complete standard mapping of this character set became possible only with the advent of Unicode 4.0, with the

encoding of certain characters peculiar to ST usage that had previously escaped the notice of standardization bodies.

Dictating aspects of actions taken in step 4, step 5 is ongoing, and addresses a whole set of

RELATED ISSUES:

(A) relative primacy and maintenance of the data in original and converted forms;

(B) database application data requirements;

(C) font support.

CONCLUSION

Unicode now serves as the standard interface to STEDT's lexical data, and with application and font support coming soon to a computer near you, Unicode will provide linguists worldwide with access to this valuable data for years to come.

See also:

Unicode Technical Note 19 (Deborah Anderson et al.)
<http://www.unicode.org/notes/tn19/> (forthcoming)

Linguists who are interested in establishing a standard way to transcribe a language, whether anyone has transcribed it before or not, whether it has an orthography of its own already or not, and whether this transcription is relatively narrow or broad, could benefit from reading this.

Notes on the next two slides:

These tables relate to the work presented in the following:

- "On the status of the curly-tail alveolo-palatal symbols." (PDF) <http://stedt.berkeley.edu/pdf/curly-tailed-tdnlcz.pdf>

- "Proposal to add five phonetic characters to the UCS" [with Everson]. (PDF) <http://linguistics.berkeley.edu/~rscook/pdf/UniProp-Final/n2366r-curly-tail.pdf>

Both available at <http://socrates.berkeley.edu/~rscook/html/writing.html>. See those documents for detailed discussion.

Briefly:

The first table derives from Wu Zongji (1992), exhibiting transcription symbols missing from Unicode 3.x.

The second table shows in the "A-P" column a more complete set of alveolo-palatal ("culry-tail") symbols, some of which were added to Unicode 4.0 through the document proposal "n2366r" (second item above).

	部位 / 方法			双唇	齿唇	齿间	舌尖前	舌尖后	舌叶(舌尖及面)	舌面前	舌面中	舌根(舌面后)	小舌	喉壁	喉
辅音	塞	清	不送气	p			t	ʈ		ȶ	c	k	q		ʔ
			送气	pʻ			tʻ	ʈʻ		ȶʻ	cʻ	kʻ	qʻ		ʔʻ
		浊	不送气	b			d	ɖ		ȡ	ɟ	g	G		
			送气	bʻ			dʻ	ɖʻ		ȡʻ	ɟʻ	gʻ	Gʻ		
	塞擦	清	不送气		pf	tθ	ts	tʂ	tʃ	tɕ					
			送气		pfʻ	tθʻ	tsʻ	tʂʻ	tʃʻ	tɕʻ					
		浊	不送气		bv	dð	dz	dʐ	dʒ	dʑ					
			送气		bvʻ	dðʻ	dzʻ	dʐʻ	dʒʻ	dʑʻ					
	鼻	浊		m	ɱ		n	ɳ		ȵ	ɲ	ŋ	N		
	滚	浊					r						R		
	闪	浊					ɾ	ɽ					R		
	边	浊					l	ɭ			ʎ				
	边擦	清					ɬ								
		浊					ɮ								
	擦	清		ɸ	f	θ	s	ʂ	ʃ	ɕ	ç	x	χ	ħ	h
		浊		β	v	ð	z	ʐ	ʒ	ʑ	j	ɣ	ʁ	ʕ	ɦ
	无擦通音及半元音	浊		w ｜ ɥ	ʋ			ɹ			j(ɥ)	(w)	ʁ		

元音		圆唇元音	舌尖元音 前	舌尖元音 后	舌面元音 前	舌面元音 央	舌面元音 后
	高	(ɥʮʉʷu)	ɿ ʯ	ɿ ʯ	iy	ɨʉ	ɯu
					ɪʏ		ʊ
	半高	(øo)			eø	ɵ	ɤ o
					E	ɜ	
	半低	(œɔ)			ɛœ	ɜ	ʌ ɔ
					æ	ɐ	
	低	(ɒ)			a	A	ɑ ɒ

CONSONANTS (PULMONIC AND GLOTTALIC)

L		L-D		I-D		D(A)		R		P-A		A-P		P		V		U		Ph		G	
p	b					t	d	ʈ	ɖ			ʇ	ᶑ	c	ɟ	k	g	q	ɢ			ʔ	
	m		ɱ				n		ɳ				ᶇ		ɲ		ŋ		ɴ				
	ʙ						r												ʀ				
							ɾ		ɽ				ᶉ										
ɸ	β	f	v	θ	ð	s	z	ʂ	ʐ	ʃ	ʒ	ɕ	ʑ	ç	ʝ	x	ɣ	χ	ʁ	ħ	ʕ	h	ɦ
						ɬ	ɮ																
			ʋ				ɹ		ɻ				ᶊ		j		ɰ						
							l		ɭ				ᶅ		ʎ		ʟ						
p’						t’		ʈ’				ʇ’		c’		k’		q’					
ɓ	ɓ					ɗ	ɗ					ᶑ	ᶑ	ɠ	ʄ	ƙ	ɠ	ɋ	ɠ				

Key: **L**=labio-/labial, **D**=dental, **I**=inter-, **A**=alveolo-/alveolar, **R**=retroflex, **P**=palato-/palatal, **V**=velar, **U**=uvular, **Ph**=Pharyngeal, **G**=glottal.
Consonants arranged according to place of articulation (left-to-right = front-to-back), with inclusion of alveolo-palatal (A-P) place series.
After WU Zongji (1992), with additional A-P place series symbols: [ʇ] A-P flap or tap, [ɻ] A-P approximant, [ᶅ] A-P lateral, and [ᶑ ᶑ] A-P implosives.
For a discussion this notation please see the most recent draft of the paper at <http://stedt.berkeley.edu/pdf/curly-tailed-tdnlcz.pdf>.

VOWELS

i	y			ɨ	ʉ			ɯ	u
		ɪ	ʏ				ʊ		
e	ø							ɤ	o
				ə	ɵ				
ɛ	œ							ʌ	ɔ
æ				ɐ					
a	ɶ							ɑ	ɒ

Richard S. Cook, UC Berkeley Linguistics
<rscook@socrates.berkeley.edu>

```
#!/usr/bin/perl -w
BEGIN { $^W = 1 }
use strict;

##############################################
# This  program  converts  Mac STEDTFont 5.1 #
# text files to UTF-8. Run it in a directory #
# of files and convert each file.txt in that #
# directory to file.u8, silently overwriting #
# any pre-existing file.u8 in that directory #
# (you might want to be careful about that). #
##############################################
# The STEDT Project  and  the author make no #
# expressed or implied warranty of any kind, #
# and  assume  no  liability  for  errors or #
# omissions.   No  liability is  assumed for #
# incidental  and  consequential  damages in #
# connection with  or arising out of the use #
# of the information  or  programs contained #
# in or accompanying this file. Caveat Utor! #
##############################################
# Richard Cook  rscook@socrates.berkeley.edu #
##############################################
# Written with MacPerl 5.6.1r2  (2003-09-09) #
# Updated for  Perl    5.8.1    (2004-09-17) #
##############################################

warn "x"x64,"\n";
warn "Started: ", scalar localtime(), " ...\n";
warn sprintf "Perl %vd\n", $^V;
warn "-"x64,"\n";
warn "Remapping Mac STEDT Font 5.1 text files to Unicode 4.0 ...\n";

#################################
###GET LIST OF FILES TO CONVERT###
#opendir(DIR, ':')
opendir(DIR, '.')
    or die 'cannot open : $!';
my @files = sort grep /\.txt$/, readdir(DIR);
closedir(DIR);
die "Died: No file.txt in current directory to convert! (why not add some?)\n"
    unless @files;
my %files;
warn "-"x64,"\n";
my $tif = @files;
warn "The following $tif infiles in the current directory will be converted\n";
warn "to UTF-8 (INFILE => OUTFILE):\n";
warn "-"x64,"\n";
my $n;
for my $infile (@files){
    $n++;
    my $outfile = $infile;
    $outfile =~ s/\.txt/.u8/
        or die "Died!: Infiles must be named with a `.txt' extension\n";
    $files{$infile}=$outfile;
    warn "$n:  $infile => $outfile\n";
}

########################################
###LOAD THE MAPS FROM THE DATA LIST BELOW###
```

```
warn "-"x64,"\n";
warn "Building the maps ...\n";

my (%char, %diac, %skip, %hash, %all, $linecount, $seq);
while (<DATA>){
    next if /^#|^$/;
    $linecount++;
    chomp;
    my @t = split(/\t/,$_,-1);
    @t == 5
        or die "Died! corrupt STED2U-map, line: $.\n";

    $seq = $t[0]
        unless defined $seq;
    $t[0] == $seq
        or die "Died! $t[0] == $seq ??\n";

    $t[1] = chr hex $t[1];

    chr $t[0] eq $t[1]
        or die "Died! $t[0] eq $t[1] ??\n";

    exists $all{$t[1]}
        or
    exists $hash{$t[1]}
        or
    exists $char{$t[1]}
        or
    exists $diac{$t[1]}
        and die "Non-unique keys! $_\n";

        $all{$t[1]}=$t[2];

    if($t[4] == 0){# these base chars *don't* remap
        $skip{$t[1]}=$t[2];
    }
    elsif($t[4] == 1){# diacritics
        $diac{$t[1]}=$t[2];
        $hash{$t[1]}=$t[2];
    }
    elsif($t[4] == 2){# these base chars *do* remap
        $char{$t[1]}=$t[2];
        $hash{$t[1]}=$t[2];
    }
    else{
        die "bad `Combining' value $t[4]: $_\n";
    }
    $seq++;
}
my $all = scalar keys %all;

warn "STEDTFont 5.1 Unicode 4.0 Scalar Values all loaded.\n";
warn "Total of $all characters in the comprehensive Map.\n";

###############
###MAP STATS###
warn "-"x64,"\n";
warn "Verifying map counts ...\n\n";

my %warn = (
```

```
    Char => scalar keys %char,
    Diac => scalar keys %diac,
    Skip => scalar keys %skip,
    Hash => scalar keys %hash,
);

my $z;
for my $w (sort keys %warn){
    my $warn = sprintf "Total of %03d Keys in the %s hash", $warn{$w}, $w;
    $warn = $warn . " (sum of above, these all get remapped)"
        if $w eq "Hash";
    $warn = $warn . " (these don't get remapped)"
        if $w eq "Skip";
    warn "$warn.\n";
    $z += $warn{$w}
        unless $w eq "Hash";
}

if($linecount == $z && $linecount == $all){
    warn "Total of $linecount Map DATA lines is all of Char, Diac, & Skip.\n\n";
}
else{
    die "ebod?! linecount=>$linecount,  z=>$z, all=>$all\n";
}

my $c = join("", sort keys %char);
my $d = join("", sort keys %diac);
my $s = join("", sort keys %skip);
my $h = join("", sort keys %hash);

my $cx = join("", map {sprintf "0x%X ", ord $_} sort keys %char);
my $dx = join("", map {sprintf "0x%X ", ord $_} sort keys %diac);
my $hx = join("", map {sprintf "0x%X ", ord $_} sort keys %hash);

warn "STEDTFont 5.1 characters to remap:\n\n";
warn "    Total of $warn{Char} base characters:\n[$cx].\n\n";
warn "    Total of $warn{Diac} prestruck diacritics:\n[$dx].\n\n";
warn "    Total of $warn{Hash} characters (base+diacritic) get remapped:\n[$hx].\n\n";
warn "STEDTFont 5.1 base characters unchanged:\n\n";
warn "    Total of $warn{Skip} chars in STEDTFont skipclass:\n[$s].\n\n";

warn "All map counts verified.\n";

##########################
###MAKE THE CONVERSION###
warn "-"x64,"\n";
warn "Beginning file conversion to UTF-8 ...\n\n";

# Some 7-bit control characters < 0x20 <SPACE> should probably
# not be in a Mac STEDTFont text file, and might be stripped
# if they occur, except for 0x09 <TAB>, 0x0A <LF>, 0x0D <CR>.
# Some applications might export other 7-bit controls in
# plain text to preserve certain info, e.g. FileMaker uses
# 0x0B <VT> for line breaks within a cell. Also, (the one
# control > 0x20 <SPACE>) 0x7F <DELETE> is unused in STEDTFont.
# We leave these controls in the data if they occur, but warn you
# in case they are evidence of some problem with the infile.

my $ctrl = '[\x00-\x08\x0B\x0C\x0E-\x1F\x7F]';
```

```
for my $f (sort keys %files){
    open IF, "< $f"
        or die "cannot open `$f': $!";
    warn "Reading text file `$f' ...\n";
    open OF, "> $files{$f}"
        or die "cannot create `$files{$f}': $!";
    warn "Writing text file `$files{$f}' ...\n";

    my %frq; my $warned=0;
    while (<IF>){

        if (/$ctrl/ && $warned < 1){
            warn "Check for improper controls in file `$f':\n $_";
            $warned++;
        }
        # mark controls (see above)
        #$_ =~ s/($ctrl)/"{".sprintf("%0#4X",ord $1)."}"/ego;

        for my $c (split(//,$_)){
            next if $c =~ /^$|\t|\n/;
            $frq{$c}++;
        }

        #my @t = split(/\t/,$_,-1);
        # if your infile is tab-separated with mixed encodings
        # then you'll want to specify the field to convert ...

        # interpose a space between diac and line-end or tab
        $_ =~ s/([\Q$d\E])(\t|$)/$1 $2/go;

        # then, any post-diac non-diac is base; reorder them
        $_ =~ s/([\Q$d\E]+)([^\Q$d\E])/$2$1/go;

        # remap to USV('s) to UTF-8
        $_ =~ s/([\Q$h\E])/ul($hash{$1})/ego;

        # note that 0x84 and 0x96 both have 1-to-2 mappings which
        # could be handled with 0x84 => U+0221, 0x96 => U+0236 instead ...

        print OF;

    }
    warn "line count for `$f': $. .\n";

    close IF
        or die "trouble closing `$f': $!";
    close OF
        or die "trouble closing `$files{$f}': $!";

    warn "Finished converting `$f'.\n\n";

    my $types = scalar keys %frq;
    warn "Frequencies for $types character types occurring in `$f':\n";
    for my $f (sort {$frq{$b} <=> $frq{$a}
                                or
                           $a cmp $b} keys %frq){
                next if $f =~ /^$/;
        my $g = $f;
        $g = "" if $g =~ /$ctrl/o;
                $g = $all{$f} if exists $all{$f};
```

```
        printf "%#X\t% 6d\t=> %s\n", ord $f, $frq{$f}, $g;
    }
    warn "-"x64,"\n";
}
my $pl = $tif == 1 ? "" : "s";
my $ar = "no";
$ar = "the" if $tif > 0;
$ar = "all" if $tif > 2;
my $time = time() - $^T;
my $sex = $time  == 1 ? "" : "s";
warn "Finished converting $ar $tif file$pl in a total of $time second$sex.\n";

#################
###SUBROUTINES###

##################################
############# SUB UL #############
##################################
###### sends  SUB U8 a list ######
###### of  bracketed  USV's ######
##################################

sub ul {
    my $x = shift @_;
    my $h = '[0-9A-Fa-f]';
    my (@l) = $x =~ /(\[U\+$h{1,6}\])/g;
    die "Yikes? no \@l ???\n"
        unless @l;
    my $u = "";
    for my $l (@l){
        $u .= u8($l);
    }
    return $u;
} # sub ul

##################################
######### SUB U8 bit ops #########
##################################
##### accepts both bracketed #####
##### and  unbracketed   USV #####
##################################

sub u8 {
    my $x = shift @_;
    my $h = '[0-9A-Fa-f]';

    #return $x unless
    #    ($x =~ s/^\[U\+($h{1,6})\]$/$1/ == 1);

    $x =~ s/^\[U\+($h{1,6})\]$/$1/;
    $x = hex($x);

    my $u8;
    if($x <= 0x0000007F){
        $u8=
        pack ("C", $x);
    }
    elsif($x <= 0x000007FF){
        $u8=
        pack ("C", 0xC0|(($x>>6))) .
```

```
        pack ("C", 0x80|($x&((1<<6)-1)));
    }
    elsif($x <= 0x0000FFFF){
        $u8=
        pack ("C", 0xE0|(($x>>12))) .
        pack ("C", 0x80|(($x>>6)&((1<<6)-1))) .
        pack ("C", 0x80|($x&((1<<6)-1)));
    }
    elsif($x <= 0x001FFFFF){
        $u8=
        pack ("C", 0xF0|(($x>>18))) .
        pack ("C", 0x80|(($x>>12)&((1<<6)-1))) .
        pack ("C", 0x80|(($x>>6)&((1<<6)-1))) .
        pack ("C", 0x80|($x&((1<<6)-1)));
    }
    elsif($x <= 0x03FFFFFF){
        $u8=
        pack ("C", 0xF8|(($x>>24))) .
        pack ("C", 0x80|(($x>>18)&((1<<6)-1))) .
        pack ("C", 0x80|(($x>>12)&((1<<6)-1))) .
        pack ("C", 0x80|(($x>>6)&((1<<6)-1))) .
        pack ("C", 0x80|($x&((1<<6)-1)));
    }
    elsif($x <= 0x7FFFFFFF){
        $u8=
        pack ("C", 0xFC|(($x>>30))) .
        pack ("C", 0x80|(($x>>24)&((1<<6)-1))) .
        pack ("C", 0x80|(($x>>18)&((1<<6)-1))) .
        pack ("C", 0x80|(($x>>12)&((1<<6)-1))) .
        pack ("C", 0x80|(($x>>6)&((1<<6)-1))) .
        pack ("C", 0x80|($x&((1<<6)-1)));
    }
    return $u8;
}# sub u8

####################################
#MAPPING DATA STEDTFONT5 <=> UNICODE4

__DATA__
#DEC    HEX     USV     UNAME   COMB
032     20      [U+0020]        SPACE   0
033     21      [U+0021]        EXCLAMATION MARK        0
034     22      [U+0022]        QUOTATION MARK  0
035     23      [U+0023]        NUMBER SIGN     0
036     24      [U+1D4A]        MODIFIER LETTER SMALL SCHWA     2
037     25      [U+0279]        LATIN SMALL LETTER TURNED R     2
038     26      [U+2AA4]        GREATER-THAN OVERLAPPING LESS-THAN      2
039     27      [U+0027]        APOSTROPHE      0
040     28      [U+0028]        LEFT PARENTHESIS        0
041     29      [U+0029]        RIGHT PARENTHESIS       0
042     2A      [U+002A]        ASTERISK        0
043     2B      [U+002B]        PLUS SIGN       0
044     2C      [U+002C]        COMMA   0
045     2D      [U+002D]        HYPHEN-MINUS    0
046     2E      [U+002E]        FULL STOP       0
047     2F      [U+002F]        SOLIDUS 0
048     30      [U+0030]        DIGIT ZERO      0
049     31      [U+0031]        DIGIT ONE       0
050     32      [U+0032]        DIGIT TWO       0
051     33      [U+0033]        DIGIT THREE     0
```

```
052     34      [U+0034]        DIGIT FOUR      0
053     35      [U+0035]        DIGIT FIVE      0
054     36      [U+0036]        DIGIT SIX       0
055     37      [U+0037]        DIGIT SEVEN     0
056     38      [U+0038]        DIGIT EIGHT     0
057     39      [U+0039]        DIGIT NINE      0
058     3A      [U+003A]        COLON   0
059     3B      [U+003B]        SEMICOLON       0
060     3C      [U+003C]        LESS-THAN SIGN  0
061     3D      [U+003D]        EQUALS SIGN     0
062     3E      [U+003E]        GREATER-THAN SIGN       0
063     3F      [U+003F]        QUESTION MARK   0
064     40      [U+032A]        COMBINING BRIDGE BELOW  1
065     41      [U+0041]        LATIN CAPITAL LETTER A  0
066     42      [U+0042]        LATIN CAPITAL LETTER B  0
067     43      [U+0043]        LATIN CAPITAL LETTER C  0
068     44      [U+0044]        LATIN CAPITAL LETTER D  0
069     45      [U+0045]        LATIN CAPITAL LETTER E  0
070     46      [U+0046]        LATIN CAPITAL LETTER F  0
071     47      [U+0047]        LATIN CAPITAL LETTER G  0
072     48      [U+0048]        LATIN CAPITAL LETTER H  0
073     49      [U+0049]        LATIN CAPITAL LETTER I  0
074     4A      [U+004A]        LATIN CAPITAL LETTER J  0
075     4B      [U+004B]        LATIN CAPITAL LETTER K  0
076     4C      [U+004C]        LATIN CAPITAL LETTER L  0
077     4D      [U+004D]        LATIN CAPITAL LETTER M  0
078     4E      [U+004E]        LATIN CAPITAL LETTER N  0
079     4F      [U+004F]        LATIN CAPITAL LETTER O  0
080     50      [U+0050]        LATIN CAPITAL LETTER P  0
081     51      [U+0051]        LATIN CAPITAL LETTER Q  0
082     52      [U+0052]        LATIN CAPITAL LETTER R  0
083     53      [U+0053]        LATIN CAPITAL LETTER S  0
084     54      [U+0054]        LATIN CAPITAL LETTER T  0
085     55      [U+0055]        LATIN CAPITAL LETTER U  0
086     56      [U+0056]        LATIN CAPITAL LETTER V  0
087     57      [U+0057]        LATIN CAPITAL LETTER W  0
088     58      [U+0058]        LATIN CAPITAL LETTER X  0
089     59      [U+0059]        LATIN CAPITAL LETTER Y  0
090     5A      [U+005A]        LATIN CAPITAL LETTER Z  0
091     5B      [U+005B]        LEFT SQUARE BRACKET     0
092     5C      [U+0300]        COMBINING GRAVE ACCENT  1
093     5D      [U+005D]        RIGHT SQUARE BRACKET    0
094     5E      [U+027E]        LATIN SMALL LETTER R WITH FISHHOOK      2
095     5F      [U+0331]        COMBINING MACRON BELOW  1
096     60      [U+00D0]        LATIN CAPITAL LETTER ETH        2
097     61      [U+0061]        LATIN SMALL LETTER A    0
098     62      [U+0062]        LATIN SMALL LETTER B    0
099     63      [U+0063]        LATIN SMALL LETTER C    0
100     64      [U+0064]        LATIN SMALL LETTER D    0
101     65      [U+0065]        LATIN SMALL LETTER E    0
102     66      [U+0066]        LATIN SMALL LETTER F    0
103     67      [U+0067]        LATIN SMALL LETTER G    0
104     68      [U+0068]        LATIN SMALL LETTER H    0
105     69      [U+0069]        LATIN SMALL LETTER I    0
106     6A      [U+006A]        LATIN SMALL LETTER J    0
107     6B      [U+006B]        LATIN SMALL LETTER K    0
108     6C      [U+006C]        LATIN SMALL LETTER L    0
109     6D      [U+006D]        LATIN SMALL LETTER M    0
110     6E      [U+006E]        LATIN SMALL LETTER N    0
111     6F      [U+006F]        LATIN SMALL LETTER O    0
```

```
112     70      [U+0070]        LATIN SMALL LETTER P    0
113     71      [U+0071]        LATIN SMALL LETTER Q    0
114     72      [U+0072]        LATIN SMALL LETTER R    0
115     73      [U+0073]        LATIN SMALL LETTER S    0
116     74      [U+0074]        LATIN SMALL LETTER T    0
117     75      [U+0075]        LATIN SMALL LETTER U    0
118     76      [U+0076]        LATIN SMALL LETTER V    0
119     77      [U+0077]        LATIN SMALL LETTER W    0
120     78      [U+0078]        LATIN SMALL LETTER X    0
121     79      [U+0079]        LATIN SMALL LETTER Y    0
122     7A      [U+007A]        LATIN SMALL LETTER Z    0
123     7B      [U+007B]        LEFT CURLY BRACKET      0
124     7C      [U+0301]        COMBINING ACUTE ACCENT  1
125     7D      [U+007D]        RIGHT CURLY BRACKET     0
126     7E      [U+0303]        COMBINING TILDE 1
127     7F      [U+007F]        DELETE  0
128     80      [U+00BB]        RIGHT-POINTING DOUBLE ANGLE QUOTATION MARK      2
129     81      [U+0252]        LATIN SMALL LETTER TURNED ALPHA 2
130     82      [U+025F]        LATIN SMALL LETTER DOTLESS J WITH STROKE        2
131     83      [U+0307]        COMBINING DOT ABOVE     1
132     84      [U+0064][U+0311]        <LATIN SMALL LETTER D, COMBINING INVERTED BREVE>        2
133     85      [U+02B7]        MODIFIER LETTER SMALL W 2
134     86      [U+0335]        COMBINING SHORT STROKE OVERLAY  1
135     87      [U+00E6]        LATIN SMALL LETTER AE   2
136     88      [U+02C7]        CARON   2
137     89      [U+02C6]        MODIFIER LETTER CIRCUMFLEX ACCENT       2
138     8A      [U+00AB]        LEFT-POINTING DOUBLE ANGLE QUOTATION MARK       2
139     8B      [U+033E]        COMBINING VERTICAL TILDE        1
140     8C      [U+0251]        LATIN SMALL LETTER ALPHA        2
141     8D      [U+00E7]        LATIN SMALL LETTER C WITH CEDILLA       2
142     8E      [U+0323]        COMBINING DOT BELOW     1
143     8F      [U+1D07]        LATIN LETTER SMALL CAPITAL E    2
144     90      [U+24D2]        CIRCLED LATIN SMALL LETTER C    2
145     91      [U+0250]        LATIN SMALL LETTER TURNED A     2
146     92      [U+031B]        COMBINING HORN  1
147     93      [U+1D2E]        MODIFIER LETTER CAPITAL B       2
148     94      [U+0131]        LATIN SMALL LETTER DOTLESS I    2
149     95      [U+0268]        LATIN SMALL LETTER I WITH STROKE        2
150     96      [U+0074][U+0311]        <LATIN SMALL LETTER T, COMBINING INVERTED BREVE>        2
151     97      [U+0153]        LATIN SMALL LIGATURE OE 2
152     98      [U+0275]        LATIN SMALL LETTER BARRED O     2
153     99      [U+00B7]        MIDDLE DOT      2
154     9A      [U+00F8]        LATIN SMALL LETTER O WITH STROKE        2
155     9B      [U+00D8]        LATIN CAPITAL LETTER O WITH STROKE      2
156     9C      [U+24B8]        CIRCLED LATIN CAPITAL LETTER C  2
157     9D      [U+1D2C]        MODIFIER LETTER CAPITAL A       2
158     9E      [U+24C8]        CIRCLED LATIN CAPITAL LETTER S  2
159     9F      [U+0303]        COMBINING TILDE 1
160     A0      [U+0236]        LATIN SMALL LETTER T WITH CURL  2
161     A1      [U+02F0]        MODIFIER LETTER LOW UP ARROWHEAD        2
162     A2      [U+2074]        SUPERSCRIPT FOUR        2
163     A3      [U+00B3]        SUPERSCRIPT THREE       2
164     A4      [U+2076]        SUPERSCRIPT SIX 2
165     A5      [U+2078]        SUPERSCRIPT EIGHT       2
166     A6      [U+2077]        SUPERSCRIPT SEVEN       2
167     A7      [U+0329]        COMBINING VERTICAL LINE BELOW   1
168     A8      [U+0282]        LATIN SMALL LETTER S WITH HOOK  2
169     A9      [U+0263]        LATIN SMALL LETTER GAMMA        2
170     AA      [U+00B2]        SUPERSCRIPT TWO 2
171     AB      [U+0309]        COMBINING HOOK ABOVE    1
```

172 AC [U+02EF] MODIFIER LETTER LOW DOWN ARROWHEAD 2
173 AD [U+25E6] WHITE BULLET 2
174 AE [U+02BB] MODIFIER LETTER TURNED COMMA 2
175 AF [U+028C] LATIN SMALL LETTER TURNED V 2
176 B0 [U+2075] SUPERSCRIPT FIVE 2
177 B1 [U+007E] TILDE 2
178 B2 [U+0235] LATIN SMALL LETTER N WITH CURL 2
179 B3 [U+014B] LATIN SMALL LETTER ENG 2
180 B4 [U+0285] LATIN SMALL LETTER SQUAT REVERSED ESH 2
181 B5 [U+0271] LATIN SMALL LETTER M WITH HOOK 2
182 B6 [U+0221] LATIN SMALL LETTER D WITH CURL 2
183 B7 [U+026F] LATIN SMALL LETTER TURNED M 2
184 B8 [U+03B2] GREEK SMALL LETTER BETA 2
185 B9 [U+0278] LATIN SMALL LETTER PHI 2
186 BA [U+0253] LATIN SMALL LETTER B WITH HOOK 2
187 BB [U+2079] SUPERSCRIPT NINE 2
188 BC [U+2070] SUPERSCRIPT ZERO 2
189 BD [U+027F] LATIN SMALL LETTER REVERSED R WITH FISHHOOK 2
190 BE [U+02B0] MODIFIER LETTER SMALL H 2
191 BF [U+0254] LATIN SMALL LETTER OPEN O 2
192 C0 [U+0266] LATIN SMALL LETTER H WITH HOOK 2
193 C1 [U+00B9] SUPERSCRIPT ONE 2
194 C2 [U+026C] LATIN SMALL LETTER L WITH BELT 2
195 C3 [U+24C1] CIRCLED LATIN CAPITAL LETTER L 2
196 C4 [U+03B8] GREEK SMALL LETTER THETA 2
197 C5 [U+03C7] GREEK SMALL LETTER CHI 2
198 C6 [U+0283] LATIN SMALL LETTER ESH 2
199 C7 [U+030C] COMBINING CARON 1
200 C8 [U+0302] COMBINING CIRCUMFLEX ACCENT 1
201 C9 [U+221A] SQUARE ROOT 2
202 CA [U+00A0] NO-BREAK SPACE 2
203 CB [U+1D00] LATIN LETTER SMALL CAPITAL A 2
204 CC [U+026E] LATIN SMALL LETTER LEZH 2
205 CD [U+007C] VERTICAL LINE 2
206 CE [U+0300] COMBINING GRAVE ACCENT 1
207 CF [U+0309] COMBINING HOOK ABOVE 1
208 D0 [U+0328] COMBINING OGONEK 1
209 D1 [U+0304] COMBINING MACRON 1
210 D2 [U+032F] COMBINING INVERTED BREVE BELOW 1
211 D3 [U+0306] COMBINING BREVE 1
212 D4 [U+0325] COMBINING RING BELOW 1
213 D5 [U+030A] COMBINING RING ABOVE 1
214 D6 [U+0294] LATIN LETTER GLOTTAL STOP 2
215 D7 [U+24CB] CIRCLED LATIN CAPITAL LETTER V 2
216 D8 [U+028F] LATIN LETTER SMALL CAPITAL Y 2
217 D9 [U+0308] COMBINING DIAERESIS 1
218 DA [U+02B1] MODIFIER LETTER SMALL H WITH HOOK 2
219 DB [U+0346] COMBINING BRIDGE ABOVE 1
220 DC [U+027D] LATIN SMALL LETTER R WITH TAIL 2
221 DD [U+025A] LATIN SMALL LETTER SCHWA WITH HOOK 2
222 DE [U+02B4] MODIFIER LETTER SMALL TURNED R 2
223 DF [U+02BC] MODIFIER LETTER APOSTROPHE 2
224 E0 [U+0280] LATIN LETTER SMALL CAPITAL R 2
225 E1 [U+0301] COMBINING ACUTE ACCENT 1
226 E2 [U+0234] LATIN SMALL LETTER L WITH CURL 2
227 E3 [U+028B] LATIN SMALL LETTER V WITH HOOK 2
228 E4 [U+025B] LATIN SMALL LETTER OPEN E 2
229 E5 [U+0290] LATIN SMALL LETTER Z WITH RETROFLEX HOOK 2
230 E6 [U+0288] LATIN SMALL LETTER T WITH RETROFLEX HOOK 2
231 E7 [U+0309] COMBINING HOOK ABOVE 1

```
232     E8      [U+1D1C]        LATIN LETTER SMALL CAPITAL U    2
233     E9      [U+026A]        LATIN LETTER SMALL CAPITAL I    2
234     EA      [U+02CC]        MODIFIER LETTER LOW VERTICAL LINE       2
235     EB      [U+0256]        LATIN SMALL LETTER D WITH TAIL  2
236     EC      [U+00F0]        LATIN SMALL LETTER ETH  2
237     ED      [U+0264]        LATIN SMALL LETTER RAMS HORN    2
238     EE      [U+1D34]        MODIFIER LETTER CAPITAL H       2
239     EF      [U+0292]        LATIN SMALL LETTER EZH  2
240     F0      [U+0291]        LATIN SMALL LETTER Z WITH CURL  2
241     F1      [U+1D38]        MODIFIER LETTER CAPITAL L       2
242     F2      [U+02D0]        MODIFIER LETTER TRIANGULAR COLON        2
243     F3      [U+0265]        LATIN SMALL LETTER TURNED H     2
244     F4      [U+0281]        LATIN LETTER SMALL CAPITAL INVERTED R   2
245     F5      [U+0111]        LATIN SMALL LETTER D WITH STROKE        2
246     F6      [U+0274]        LATIN LETTER SMALL CAPITAL N    2
247     F7      [U+1D39]        MODIFIER LETTER CAPITAL M       2
248     F8      [U+0272]        LATIN SMALL LETTER N WITH LEFT HOOK     2
249     F9      [U+0273]        LATIN SMALL LETTER N WITH RETROFLEX HOOK        2
250     FA      [U+0259]        LATIN SMALL LETTER SCHWA        2
251     FB      [U+0255]        LATIN SMALL LETTER C WITH CURL  2
252     FC      [U+0300]        COMBINING GRAVE ACCENT  1
253     FD      [U+21AD]        LEFT RIGHT WAVE ARROW   2
254     FE      [U+21AE]        LEFT RIGHT ARROW WITH STROKE    2
255     FF      [U+0301]        COMBINING ACUTE ACCENT  1
#END
```

BEGIN ABSTRACT

Linguistic Society of America (LSA) 2005, Oakland, CA

 http://www.lsadc.org/annmeet/sessions.html#sunmorn
 Workshop: Unicode for Linguists: How To Type, Send, and Archive
 Linguistic Texts by Computer
 Sunday, 9 January 9:00 AM - 12:00 PM
 Room: Simmons 3/4, Oakland Convention Center.
 Organizer: Deborah Anderson (UC-Berkeley)
 Participants: Charles A. Bigelow (Bigelow & Holmes), William Bright,
 Peter Constable (Microsoft), Richard Cook (UC-Berkeley), Kenneth
 Whistler (Sybase, Inc.)

--- Workshop Schedule ---

 9:00 - 9:05   Introductory remarks by William Bright

 9:05 - 9:50   "Introduction to Unicode for Linguists"
                by Peter Constable

 9:55 - 10:25   "Unicode Fonts for Linguists (with a demonstration)"
                by Charles Bigelow

 10:30 - 11:00 "Conversion of Legacy Linguistic Transcription data to
                Unicode 4.0" by Richard Cook

 11:05 - 11:20 "Developing Tools with IPA-encoded Unicode: Towards a
                Phonologically-Based Search Engine" by Edward Garrett

 11:25 - 11:40 "The Future of Unicode"
                by Deborah Anderson and Ken Whistler

 11:40 - 12:00 Discussion and Q & A period with panel participants.

COOK

 Dr. Richard Cook (Project Manager and Systems Administrator, the
 Sino-Tibetan Etymological Dictionary and Thesaurus at UC Berkeley
 <http://stedt.berkeley.edu>; Unicode <http://www.unicode.org/>
 editorial committee member, representative to ISO/IEC
 JTC1/SC2/WG2/IRG <http://www.cse.cuhk.edu.hk/~irg/>; co-author of the
 CDL specification <http://www.wenlin.com/cdl/>; Post-Doctoral
 Researcher and programmer for the World Color Survey
 <http://www.icsi.berkeley.edu/wcs/>.)

 Richard Cook will discuss Unicode 4.0 linguistic transcription
 support, and conversion of legacy data to Unicode 4.0. The use of
 custom(izable) tools for converting legacy data to Unicode 4.0 will
 be demonstrated.

Title:

        Conversion of legacy linguistic transcription data to Unicode 4.0

Abstract:

        This presentation describes the process of converting legacy data to
 Unicode encoding, including character set mapping, encoding unencoded
 characters, and demonstration of conversion tool use. Terminology is
 introduced in context, including the following: legacy data, code

point, encoding, custom encoding, standard encoding, USV, mapping, character set, encoding form.

Unicode conversion of the STEDT Project's legacy data serves as a specific example. The STEDT Project (federally funded at UC Berkeley since 1987) began migrating its million-record Sino-Tibetan (ST) lexical relational database system to Unicode in earnest in the late 90's. The justification for this conversion hinges on attaining universal permanent data access. STEDT data was originally input and archived using a custom-encoded Apple Macintosh character set refined over the years on the basis of the transcription characters appearing in specific lexical print (and handwritten) sources. Data input using other encodings was migrated to this custom encoding. Source transcriptions would sometimes be normalized in order to render them in the custom encoding, though in general the character set was well-suited to capturing ST transcriptional conventions. The custom-encoding was not, however, well-suited to smooth data interchange and archiving.

The conversion process involved the following steps: (0) Realizing that it needed to be done, learning how to do it, and that it could only be done in stages; (1) Mapping to Unicode 3.0; (2) Formal proposal of unencoded characters for inclusion in Unicode/ISO 10646; (3) Mapping to Unicode 4.0; (4) Creation of tools for conversion of the custom-encoded data; (5) Creation of tools for use of the new Unicode data. Complete standard mapping of this character set became possible only with the advent of Unicode 4.0, with the encoding of certain characters peculiar to ST usage that had previously escaped the notice of standardization bodies.

Dictating aspects of actions taken in step 4, step 5 is ongoing, and addresses a whole set of related issues, including: (A) relative primacy and maintenance of the data in original and converted forms; (B) database application data requirements; (C) font support.

Unicode now serves as the standard interface to STEDT's lexical data, and with application and font support coming soon to a computer near you, Unicode will provide linguists worldwide with access to this valuable data for years to come.

Dr. Richard S. Cook
STEDT Project
Linguistics Dept.
UC Berkeley
http://stedt.berkeley.edu

END