Loglan Academy Agenda 3/5/2014

Randall Holmes

March 8, 2014

The purpose of the document is to lay out a list of items we already have before us, and to introduce the first layer of my global proposal to bring Loglan into line with something like my PEG parser (Phonology).
Version note: Revised 10:13 pm 3/5/2014, enforcing no qwx and fixing a bug in PreComplex having to do with three letter djifoa followed by a final borrowing.

1 The Existing Proposals with Comments

Here find the current text of the List of Proposals document with inserts by me reflecting my opinions about the action to be taken and the urgency of taking some action.

```
Proposals before la Keugru  8/28/2013 12:43 PM laptop
(added notes about implementation work to each item).

The official name of each of these proposals is of the form Proposal n, 2013.

Proposal 1 2013 passed and documented


--------------------------------------------------------------------------------


Proposal 2:  (Randall Holmes):  Ban repeated vowels aa/ee/oo in borrowings
.  Revise alkooli (the only example of such a borrowing I have found)
```

```
to alkoholi.

This proposal has PASSED.  It is recorded in Appendix H;
I am still working on
my dictionary updating skills.

Proposal 2 is implemented in my parser;
it is not yet implemented in the officially
posted dictionary though a suitably corrected copy is available.
(I think it is corrected now)

--------------------------------------------------------------------------------

Proposal 3: (John Cowan):  Introduce a word ZAO
 which when placed between
predicates has the same effect as complex formation,
 and abandon
the attempt to form complexes using borrowings.

Proposal 3B (Randall Holmes)  Introduce ZAO
as in Cowan's proposal while
 taking no negative action
(complexes with borrowings continue to be allowed,
but ZAO is available to paraphrase
these or indeed any complexes).

Proposal 3 should shortly be implemented in my parser.
```

This proposal (in the 3B version) is implemented in my current PEG grammar, and ratification will be urged as part of one of the future components of my grand proposal to bring things into line with the PEG. (added later: I am wrong; this is part of the current part of the grand proposal for accidental reasons, though not an essential part).

Proposal 4: (John Cowan) superseded, see Proposal 7 below

Proposal 5: (Randall Holmes) Eliminate noka
and all similar words.

NOTE: I think this may not be a grammatical
proposal at all but a correction of an error in
the dictionary. The action to be taken is the same:
remove the dictionary entries and eliminate this
option from the lexer.

My parser still allows formation of noka words,
but I believe it would be very hard to get the
parser to read one (and the same is true for LIP).
I should revise the parser to eliminate them.

I do not think this proposal requires any particular action, because I think it is a mistake in the dictionary. I do not think there is much danger of either my parser or LIP ever thinking that it is reading such a word. I will correct my parser (if I have not already) so that it does not recognize such words.

Proposal 6 (John Cowan): Eliminate the djifoa (affixes)
with the repeated vowels aa/ee/oo and do the required
dictionary work to rebuild affected complexes.
[he has suggested a more limited proposal to
eliminate the EE and OO djifoa]

My parser does not implement this.
It would require massive dictionary work.
A revised version leaving the AA djifoa
would have a more modest impact.

I am not against working on this proposal (perhaps think about eliminating the few EE and OO djifoa, but I think the AA djifoa are too numerous and widely used. In spirit, I agree with John, but this is one of those charming features the language is already committed to.

Proposal 7 (John Cowan -- revised to incorporate Proposal 4 text):

(a) The sounds of "x", "q", and "w" to be removed from Loglan. They are permitted only in names, and are relatively low-frequency sounds in the world's languages.

(b) The letter "h" to be allowed with either IPA /h/ (its current sound) or IPA /x/ (the current sound of "x"). This will make life easier for Spanish, Russian, and Chinese loglanists, who have /x/ in their languages but not /h/. (Hindi, English, and Japanese have /h/ only, German has both, French has neither.)

(c) Extension of "gao". Currently it is permitted only before "Ceo" and "Vfi" words to make Greek upper case letters. It is to be permitted before any phonological word to make a new word of nurcmapua TAI.

(d) Specific new words of TAI to be added to the dictionary: "gaohei" = x, "gaohai" = X, "gaokei" = q, "gaokai" = Q, "gaovei" = w, "gaovai" = W, "gao,alef" = ? (Hebrew letter alef). These replace "xei", "xai", "qei", "qai", "wsi", "wma", and nothing respectively.

[replaces original proposals 4 and 7]

My parser already makes it impossible to use w;
it treats q and x like any other consonant,
but they would be easy to excise.

I agree with this proposal, with a proviso. I do think that we need CVV words for the Latin letters thus eliminated from the alphabet. They occur commonly in mathematics and in foreign words.

I agree that x,q,w should be eliminated, but I want CVV words for at least lowercase versions of these letters.

```
Proposal 8 (Randall Holmes):  A predunit
appearing in a name must be prefixed with CI.
Rescind the earlier decision that we have an
additional pause phoneme used only in serial names.

rationale:  very simple:  this makes
La Djan, blanu a sentence rather than a name again,
and without multiple grades of pauses.

cautions:  make sure there are no
ambiguities with existing uses of CI.

La Djan, blanu once again means "John is blue".

La Djan, ci blanu, mrenu  becomes "John the Blue is a man".
(yes, the pause works to mark the predicate,
though this may not be a good practice).

My parser implements this.  This makes an
actual incompatibility between my parser and LIP;
there are things which each parses which
the other does not parse, as predunits are put into
serial names in incompatible ways.

--------------------------------------------------------------------------------
```

This one is implemented in my parser and I would like to see it ratified promptly. Getting rid of multiple pause phonemes and saving sentences like `La Djan, blanu` are important issues.

Proposal 9 2013 passed and documented

Proposal 10 (John Cowan)

The Loglan Project uses the terms
"affix" and "lexeme" in ways that
contradict standard linguistic usage.
Our complexes are composed entirely
of affixes, but an affix to a linguist is
 either a suffix or a prefix:
there must be a root to which they
are attached. I suggest we switch to
the neutral term "combining form"
until we have a Loglan term analogous
to Lojban "rafsi".

Similarly, a "lexeme" is not
 a word class based on syntactic
interchangeability, but one based
on sharing an underlying form to which
different inflections are added.
Thus "run", "runs", "running", "ran"
are all members of the "run" lexeme
in English ("runner" is not, as the
"-er" ending derives a new lexeme).
 We should instead use "nurcmapua",
X is a little-word class including Y.

Of course, this applies only
 to formal proposals and documentation and
where clarity is needed, not to casual loglandic chitchat.

Comment (Randall Holmes) This proposal
ties into my program of developing a full
Loglan vocabulary for our own grammar.
The grammar terms should have English translations

```
that a linguist would understand (and possibly
 alternative English translations which are traditional
in the L community but misleading for linguists
, and labelled as such).  An implementation
of Proposal 10 might be part of an implementation
 of the Loglan grammar terminology project.
```

I would say that this metalevel proposal has in effect been implemented (and thank you John). Please continue virtuously saying "djifoa" or "combining form" instead of "affix" except when alluding to historical documents, fellow logli!

```
Proposal 11 (Randall Holmes)

I hereby officially suggest the introduction of an infix -zie-
which can be used to merge PA class operators with
A-zie-B meaning roughly A-and then-B  or "proceeding from A to B"

then replacing each of the compound location operators with
a -zie- form

that is, vuva would be replaced by vuzieva

The rationale has been discussed:  it is part of the general program
of eliminating structure word breaks.

vu, va preda really cannot be construed as meaning the same thing as
vuva preda

My parser does not as yet implement this.
It would require modest dictionary work
to change the compound location operators.
```

This is not high on my priority list but it is needed eventually (or something like it). It will appear as part of the Lexer layer of my grand proposal.

```
Proposal 12 (John Cowan)

Currently, we have NAHU compounds for every NA word which create time,
place, and manner questions.  Grammatically these are freemods, which
means they can appear almost anywhere.  I think it would be sufficient
to treat these just as regular tagged arguments.  "Na hu" as two words
would mean "at the same time as what?" which is entirely synonymous with
"nahu" meaning "when?"

The only downside is that sentences like "I tu sonli nahu dzoru?", which
is one way of saying "When do you sleepwalk?", would have to have the
"nahu" moved to somewhere else in the sentence.  However, this is only
a trivial syntactic change; there is no semantic benefit to having it
between two predunits.

My parser does not implement this.
It would be an easy grammar change and
would involve no dictionary work, as eliminated
 words remain valid phrases with different
grammar and the same meaning.

--------------------------------------------------------------------------
```

I would say that John is right here, though I have not attended to it.

```
Proposal 13 (Randall Holmes):  A change to jelink and juelink.

JE and JUE can currently only be followed by arguments;
it should also be permissible to allow them to be followed
by modifiers

the rule should be changed to
```

```
jelink <- JE term from jelink <- JE argument

(leaving out freemods for clarity)

This is clearly grammatically harmless and allows much
finer use of modifiers (PA clauses).

examples

le mrenu je vi la hasfa bi la Djan

The man who is at the house is John

le bilti je vi lo cutri, nirli ga gudbi sucmi.

The beautiful-in-the-water girl swims well

Notice that this allows tight application of modifier clauses
as here in metaphors.

This interacts with John's proposal 12, restoring
a lot of the freedom of placement of nahu if it
becomes a modifier instead of a freemod.

I think that JEPA and JUEPA will feel like
new classes of words, though there is
no need to add them to the grammar:

one is likely to write "le mrenu jevi la hasfa".

My parser implements this.
```

I would like it if this were ratified reasonably promptly (so I suppose I am in favor of ratifying the previous one at the same time); it is already in my provisional grammar.

1.1 Further Recent Proposals

If I have forgotten someone else's additional proposal, prod me.

Proposal 14 2013: clean up uses of MO (John Cowan): It was pointed out that homonymous uses of `mo` create endless opportunities for LW breaks which must be marked by pauses. I implemented this by eliminating the `-mo` letter construction completely and replacing the 000 numeral with `moa` in my parser.

Proposal 1 2014: introduce SIE: I propose the introduction of a new word `sie` expressing apology rather than mere regret: `uu` currently expresses both, and it is an important distinction to draw. I run into this problem in speech in English frequently and I have encountered it in Loglan.

If one says `Sie` by itself (I'm sorry) I think that `Siu` would be an appropriate response (rather in the spirit of "don't mention it", which is also a phrase which can be used in place of "you are welcome", the current official translation of `siu`.

I'm very fond of this very modest proposal: I would like to see it ratified.

Proposal 2 2014: eliminate vowel-initial letterals: The vowel-initial letterals are a pain. They create the only situation where CVV-V occurs in compound little words (in acronyms, and strictly speaking this will not be entirely eliminated) and they appear to require an additional clause in the formation of predicates to handle compounds like X-ray with the letter a vowel (A-train). I modestly propose that we introduce CVV letterals for vowels. ZIA, ZIE, ZII, ZIO, ZIU are free. One might want the ZUv series as well for upper case. We could then eliminate all the vowel initial letterals and the need for special rules in various situations. I would assume one would keep the ability to abbreviate vowels in acronyms.

Note that one would not want to use -zie- as the linker for compound location operators in this case.

This proposal presents difficulties (I know it); I would like to hear discussion.

2 Holmes Program Grand Proposal Part I: Phonology

This section is an official Proposal. Details are most certainly open to discussion.

The overall Grand Proposal will be divided into three sections, Phonology, Lexicology, and Grammar. This is the Phonology section. My grammar differs from the LIP grammar in being a complete formal grammar from the level of letters upward.

The scope of this section of the proposal is the definition of the various classes of words on a purely phonological level. It should be noted that the general definition of the class of cmapua (structure words) plays an extremely limited role in the grammar: it is only used to define the sorts of things which can follow the one word quote `liu` (a fact not reflected in this section, but easily seen in the full grammar).

I include PEG expressions as formal definitions of grammatical dicta stated in English. The PEG expressions have value as giving *mathematically precise* definitions of the intended classes of strings, though of course they might possibly have bugs in them. Part of the dialogue about this proposal should be teasing out where I should be saying things more carefully or cleverly in English to signal my intentions (and where I should be saying things in English at length where I have just presented PEG gobbledygook, do not hesitate to criticize!).

```
lowercase <- ![qwx][a-z]

uppercase <- ![QWX][A-Z]

letter <- ![qwxQWX][A-Za-z]

V <- [AEIOUWYaeiouwy]

V2 <- [AEIOUaeiou]

C <- (!(V) letter)
```

A Loglan letter is any letter of the Latin alphabet other than `q, w, x`. (The elimination of `q,w,x` is a change, now implemented).

The vowels are `a,e,i,o,u` and the special vowel `y`.

The remaining letters are consonants. `m,n,l,r` admit vocalic uses.

The precise phonetic value of the Loglan letters is not discussed here as it is not a formal grammatical issue; our views are in line with existing Loglan documentation on this, with the exception of elimination of the three letters singled out above. We do note that the phonetic value of `x` is then freed up as an alternative value of `h`, and this is recommended in some contexts.

```
Mono <- ('ao' / (V2 [i]) / ([Ii] V2) / ([Uu] V2))

CVVSyll <- (C Mono)

LWunit <- ((CVVSyll V2) / (C V2 V2) / (C V2))

LW1 <- ((V2 V2) / (C V2 V2) / (C V2))
```

The pairs of regular vowels which must be pronounced as monosyllables are `ao`, `ai`, `ei`, `oi`. Any pair of regular vowels in which the first is `i` or `u` is permitted to be pronounced monosyllabically. It should be noted that the parser *always* chooses the monosyllabic pronunciation by preference. I believe there are situations in which the optional monosyllabic pairs **must** be pronounced monosyllabically, and there are no situations in which they are forced to be pronounced as disyllables.

The CVV syllables can then be identified.

The class LWunit lists the C-initial phonetic elements which can make up compound little words as described in NB3. The phonological definition of a compound little word comes into play in this grammar **only** in considering what can follow the one-word quotation word `liu`.

The class LW1 lists the building blocks of cmapua (little words) other than a V syllable which can only occur initially. Note that the CVV units in class LW1 may be disyllables.

```
caprule <- ((uppercase / lowercase) (lowercase)* !(letter))
```

This rule expresses the Loglan capitalization rule: a word (unbroken block of letters) begins with an uppercase or lowercase letter and all subsequent letters are lowercase.

```
InitialCC <- ('bl' / 'br' / 'ck' / 'cl' / 'cm' / 'cn'
/ 'cp' / 'cr' / 'ct' / 'dj' / 'dr' / 'dz' / 'fl' / 'fr' / 'gl'
/ 'gr' / 'jm' / 'kl' / 'kr' / 'mr' / 'pl' / 'pr' / 'sk' / 'sl'
/ 'sm' / 'sn' / 'sp' / 'sr' / 'st' / 'tc' / 'tr' / 'ts' / 'vl'
 / 'vr' / 'zb' / 'zv' / 'zl' / 'sv' / 'Bl' / 'Br' / 'Ck'
 / 'Cl' / 'Cm' / 'Cn' / 'Cp' / 'Cr' / 'Ct' / 'Dj' /
'Dr' / 'Dz' / 'Fl' / 'Fr' / 'Gl' / 'Gr' / 'Jm' / 'Kl'
/ 'Kr' / 'Mr' / 'Pl' / 'Pr' / 'Sk' / 'Sl' / 'Sm' / 'Sn'
 / 'Sp' / 'Sr' / 'St' / 'Tc' / 'Tr' / 'Ts' / 'Vl' / 'Vr'
/ 'Zb' / 'Zv' / 'Zl' / 'Sv')
```

Above find the list of permitted word- (and syllable-) initial consonant pairs. This list corrects two errors in L1: `sv` and `zl` must be included as they are represented in the dictionary.

```
NonmedialCC <- ('bb' / 'cc' / 'dd' / 'ff' / 'gg'
 / 'hh' / 'jj' / 'kk' / 'll' / 'mm' / 'nn' / 'pp' / 'qq'
/ 'rr' / 'ss' / 'tt' / 'vv' / 'xx' / 'zz' / ([h] C) /
 ([cjsz] [cjsz]) / 'fv' / 'kg' / 'pb' / 'td' /
([fkpt] [jz]) / 'bj' / 'sb')

NonjointCCC <- ('cdz' / 'cvl' / 'ndj' / 'ndz'
 / 'dcm' / 'dct' / 'dts' / 'pdz' / 'gts' / 'gzb'
 / 'svl' / 'jdj' / 'jtc' / 'jts' / 'jvr' / 'tvl' / 'kdz'
/ 'vts' / 'kdz' / 'vts' / 'mzb')
```

Above find the list of pairs of consonants which may **not** appear adjacent to one another, even at a syllable boundary, and a list of forbidden triples extending this.

```
RepeatedVowel <- ('aa' / 'ee' / 'oo' / 'Aa' / 'Ee' / 'Oo')

RepeatedVocalic <- ('mm' / 'nn' / 'll' / 'rr' / 'Mm' / 'Nn' / 'Ll' / 'Rr')
```

Repeated vowels require attention because whereever these vowel pairs appear (necessarily in different syllables) one of them must receive the main stress of the word it is in.

The repeated consonants signal vocalic use of these consonants. No consonant appears doubled other than these.

```
FirstConsonants <- ((!((C C RepeatedVocalic))
&(InitialCC) (C InitialCC)) / (!((C RepeatedVocalic)) InitialCC)
/ ((!(RepeatedVocalic) C) ![y]))

FirstConsonants2 <- ((!((C C RepeatedVocalic))
&(InitialCC) (C InitialCC)) / (!((C RepeatedVocalic)) InitialCC)
/ (!(RepeatedVocalic) C))
```

These rules define what blocks of consonants may appear at the beginnings of syllables (or words). This will be a block of one to three consonants, each adjacent pair being a permitted initial pair. The first rule defines those which may appear in predicates, which cannot be followed by the special vowel `y`; the second rule is for names, in which this restriction does not apply. An initial consonant group cannot include the first letter of a vocalic consonant pair.

```
VowelSegment <- (Mono / V2 / (&(RepeatedVocalic) C))
```

This rule defines the vowel segment in a syllable. This will be a monosyllabic vowel pair, or a regular vowel, or a vocalic consonant which is followed by another occurrence of the same consonant (only the first in the pair is the vowel).

```
Syllable <- ((FirstConsonants)? VowelSegment
(FinalConsonant)? (FinalConsonant)?)

FinalConsonant <-
(!(NonmedialCC) !(NonjointCCC) !(Syllable) C)

Syllable2 <- ((FirstConsonants2)? (VowelSegment / [y])
(FinalConsonant2)? (FinalConsonant2)?)

FinalConsonant2 <-
(!(NonmedialCC) !(NonjointCCC) !(Syllable2) C)
```

It is a characteristic of the new grammar that there is a formal definition of what constitutes a syllable, and resolution of words into syllables plays an important role in the parsing of predicate and name words.

There are two versions here, one for predicates and one for names. The name version differs in allowing `y` to be used as a vowel segment.

A syllable consists of an optional first consonant sequence as described above, followed by a mandatory vowel segment, followed by one or two optional final consonants. A final consonant may not begin a forbidden pair or triple of consonants (possibly looking forward past the end of the syllable) and (very important) may not start a syllable. The new parser breaks a sequence of consonants and starts a new syllable as early as it can. For example, `Loglan` is syllabized `Lo-glan`: the `g` is not a final consonant of the first syllable because it is permitted to start a syllable there, so that is where it breaks. It is not my intention to forbid people from pronouncing it `Log-lan`, but the parser in determining whether a word is legal tries to place junctures as early as possible.

Note that as many as five consonants may occur at a syllable break (two final consonants in the preceding syllable and three initial ones in the following syllable). Of course this will only happen in borrowings.

The original sources are not at all clear about the considerations which I make precise here. The rule for initial sequences of consonants is attested in NB3 (the fact that adjacent pairs must be initials) and there are initial triples in the dictionary.

The imposition of syllable structures on names is new, but I think it

makes sense. This does have the effect that names like `Rl` which are attested must now be written `Rrl`. The rules for vocalic consonants are precisely modelled on their use as joints in borrowings.

It should also be noted that syllable structure as such operates most in the context of borrowings and names. Predicates and cmapua are broken up into djifoa or unit cmapua, some of which are disyllabic CVV forms, and the syllabification is usually but not always irrelevant (it becomes relevant in situations where the repeated vowel stress rule acts).

```
Name <- (([ ])* &(((uppercase / lowercase)
((&((lowercase lowercase)) lowercase))* C
(!(.) / ',' / &(period))))
((Syllable2)+ (!(.) / ',' / &(period))) !((([ ])* [,])))
```

We are now able to precisely state the definition of a name. A name has more than one letter, is consonant-final, is followed by a comma or other terminal punctuation mark or the end of the text, and resolves into syllables of the second kind. Additional stuff at the end prevents perverse behavior of punctuation.

We now have a section commencing the development of predicate words.

```
CV <- (C V2 !(V2))

Cfinal <- ((C [y] &(letter)) /
(!(NonmedialCC) !(NonjointCCC) C !(V2)))
```

Some final forms. CV is a final CV unit not followed by a vowel. Cfinal is a final consonant (which might include a `y` for hyphenation to following letters) and in any event not initial in a forbidden sequence of consonants (the `y` option exists to prevent this situation).

```
hyphen <- (!(NonmedialCC) !(NonjointCCC)
(([r] !([r]) !(V2)) / ([n] &([r])) / [y]) &(letter) !([y]))
```

A hyphen for avoiding phonetic disasters at djifoa boundaries. It may be an `r` not followed by an `r` or a vowel in the following unit. It may be an `n` followed by an `r` in the following unit. It may be a `y`. A hyphen is always followed by some letter, which is never a `y`.

```
CVVStressed <- (C &(RepeatedVowel) V2 !(RepeatedVowel) V2 (hyphen)?)

CVVDisyllable <- (C !(Mono) V2 V2 (hyphen)?)
```

Special forms of CVV djifoa. The first are the ones which require repeated vowel stress (the vowel is AA, EE, OO). We do not allow these to be followed by vowel initial forms beginning with the same letter.

The second is just any djifoa that must be a disyllable.

The only context where exact syllabification is important is where a repeated vowel djifoa is present and so must carry the main stress, which has to be penultimate in the word in a predicate.

```
#The restriction on repeated vowels prevents a CVV affix from forcing stress on

CVV <- (C V2 !(RepeatedVowel) V2 (hyphen)?)

CVVY <- (C V2 V2 [y])

CVC <- (C V2 Cfinal)

CVCY <- (C V2 C [y])

#The repeated vowel rule and option of a y-hyphen here
are strictly to deal with following vowel-initial borrowings in complexes

CCV <- (InitialCC V2 !(RepeatedVowel) (([y] &(letter)))?)

CCVY <- (InitialCC V2 [y])

AffixY <- (CVCY / CCVY / CVVY)
```

Here are the basic shapes of three letter djifoa. Comments in the PEG file are preserved because relevant. Notice that hyphens are supported with following djifoa, either explicitly in the CVV case or implicitly as one of the options in Cfinal for the CVC case. The fact that CCV forms may require `y` hyphens when followed by vowel initial borrowings is a new observation.

The three-letter djifoa ending in Y need to be identified for a special purpose seen below in PreComplex.

```
CCVCV <- (CCV ((CV !(letter)) / (C [y] &(letter))))

CVCCV <- (C V2 !(NonmedialCC) C ((CV !(letter)) / (C [y] &(letter))))
```

Here are the five letter djifoa: the final vowels become `y` unless word-final, and the PEG enforces this.

```
CCSyllable <- (&((C C)) FirstConsonants (Mono / V)
(FinalConsonant)? (FinalConsonant)?)

CCFinalSyllable <- ((FirstConsonants)? VowelSegment
FinalConsonant &((C / ([y] C))) (FinalConsonant)?)

CCSyllableB <- ((FirstConsonants)? &(RepeatedVocalic)
 C (FinalConsonant)? (FinalConsonant)?)
```

These are sorts of syllable whose presence in a predicate signals the presence of the required CC pair. A CCsyllable has an initial consonant sequence with more than one consonant. A CCFinalSyllable has a final consonant which is followed by another consonant (which may or may not actually be in the same syllable). A CCSyllableB has a pair of adjacent consonants because it contains a vocalic pair.

```
SyllableVV <- ((FirstConsonants)? &(RepeatedVowel) V2)
```

This detects syllables which cause the repeated vowel stress rule to come into play.

```
CW1 <- ((C V2 V2) / (C V2))

BadSequence <- ((V2 V2 V2 V2)
/ (V2 (V2)? CW1)
/ (CW1 (V2 V2)) / (CW1 CW1))
```

The BadSequence rule is designed to detect the possibility of cmapua being peeled off the front of a borrowing. I do impose a stricter limitation than JCB here: he allows arbitrarily long sequences of vowels at the head of predicate borrowings, and a single C followed by arbitrarily long sequences of vowels. I rule this out, allowing no more than three vowels (two-vowel sequences are attested in the dictionary).

```
NoBadstress <- (!(SyllableVV) Syllable)
```

Syllables not invoking the repeated vowel stress rule.

```
Borrowing <- (!((C C V2 V2 !(letter)))
 !(Name) !(BadSequence)
 !((V2 InitialCC V2))
 !(((CW1 / (V2 V2) / V2) (!(V2) Borrowing)))
!(CCSyllableB)
&((NoBadstress (NoBadstress)+))
((!((CCSyllable / CCSyllableB
/ CCFinalSyllable)) Syllable))*
(Syllable)+)
```

The quite baroque definition of a borrowing. CCVV borrowings are specifically forbidden (part of one of our proposals). Borrowings are not names. Borrowings do not begin with the sorts of sequences defined under

BadSequence above (this avoids peeling off cmapua from the front). The !((V2 InitialCC V2)) avoids a situation where a cmapua can be peeled off. !(((CW1 / (V2 V2) / V2) (!(V2) Borrowing))) avoids a situation where a cmapua can be peeled off the front of a consonant initial borrowing. A borrowing may not begin with a syllable with vocalic consonants (this saves `hidrroterapi` in the dictionary). A borrowing may not contain any syllable which invokes the repeated vowel stress rule. A borrowing must contain some syllable which includes or causes a CC pair, and it must have at least two syllables.

```
Affix <- ((Borrowing [y] &(letter))
/ CCVCV / CVCCV / CCV
/ (!((CVVStressed Affix (Affix / (Borrowing !(letter)))))
 !((CVVStressed (CVVDisyllable
 / CCVCV / CVCCV / Borrowing))) CVV)
/ (CVC &(letter)))
```

This rule describes the "affix" components of complexes (using the deprecated traditional Loglan term for these).

These are Borrowings plus a `y` hyphen (and followed by a letter), the five letter djifoa already defined, the CCV djifoa, the CVV djifoa with a restriction enforcing their possible position if they involve repeated vowels, and CVC djifoa in non-final positions.

```
#It appears that a final borrowing in a complex must be y-hyphenated

PreComplex <- ((Affix
(PreComplex / (Affix !(letter))))
/ ((CCVCV / CVCCV / (Borrowing [y])
/ (AffixY)) Borrowing !(letter)))
```

A PreComplex is an initial approximation to a Complex. It consists of an Affix followed by either a PreComplex (note the recursion) or a final Affix or a list of kinds of unit all of which end in `y` followed by a final Borrowing.

Notice that a Borrowing is **not** an Affix; the reason is that we need to enforce the restriction that a final Borrowing in a complex must be preceded by a `y` hyphen.

```
Complex <- (!(Name) !((C V2 (V2)?
(PreComplex / (CVV !(letter)) /
(Borrowing !(letter)))))
 !((C V2 (!((C C (V2 / C) V2 !(letter))) InitialCC)))
PreComplex !(letter))
```

A Complex is not a name. It cannot begin with a CV(V) which can be peeled off the front. There is a complex provision for preventing the situation where CVCC with the CC a permitted initial pair allows the CV to be peeled off the front, while still preserving the six letter complexes where this is safe.

```
Primitive <- ((CCVCV / CVCCV) !(letter))

Predicate <- (&(caprule) (Primitive / Complex / Borrowing)
 !(letter) ((([ ])+ ('zao' / 'Zao') ([ ])+ Predicate))?)
```

This completes the definition of the predicate class. A Primitive is a five letter djifoa not followed by another letter (and so not ending in `y`).

A Predicate is either a primitive, a complex or a borrowing (checked in that order; of course primitives and complexes would meet the specifications to be a borrowing, which do not include provisions to exclude the other classes).

Note that my definition includes John's suggestion of an alternative method of constructing complexes using `zao`.

I have completely (and deliberately) omitted the "A-train" situation. I do not provide for attaching a VCV literal to the front of a predicate with a `y` hyphen. It could be done, but I think it is a bizarre move best avoided. The attachment of a CVV literal is allowed, but gets no special status; there is a general freedom to choose what hyphen you use in the formal grammar, though a semantic distinction might be drawn if a letteral happened to coincide with a djifoa.

```
LW <- (&(caprule) ((((!(Predicate) V2 V2))+
/ ((!(Predicate) (V2)? ((!(Predicate) LWunit))+) / V2)))
```

Here is the actual formal definition of the class of structure words (which is hardly used at all in the grammar). This is the same as the definition in NB3. It contains provisions to avoid confusion of LWs with initial segments of following predicates, which is why it cannot appear until this point.

3 Notes toward Holmes Grand Proposal Part II: Lexing

This section is working notes, not yet an official Proposal section.

This section deals with details of Loglan that are for the most part not manifest in the previous official formal grammar. The word classes are defined by LIP using rather opaque internal representations, and there are clearly bugs. Our program is to parse Loglan from the level of letters upward, and as a result we have had to mandate exact formal definitions for these word classes, which in some cases are clearly not exactly the same as those implicit in LIP. Details will be seen below. In some cases there are serious problems, as with APA words and similar phenomena.

This is also a good place to remark on the general problem of "structure word breaks". In some cases, it is necessary to explicitly break the flow of structure words with a comma which of course represents an actual pause in the flow of speech in order to prevent an undesired interpretation. The infamous example of this sort is LEPO versus LE, PO but there are others. This grammar does **not** eliminate structure word breaks, but it is part of our eventual program to minimize or eliminate them. I am told that Lojban does not have them.

```
__LWinit <- (([ ])* !(Predicate) &(caprule))

#Used to prevent CV cmapua from absorbing a following vowel

Oddvowel <- (((V2 V2))* V2)
```

These are auxiliary gadgets controlling the lexer. The first one appears at the beginning of many rules to ensure that what looks like a structure word is not in fact the beginning of a predicate (it also incidentally removes whitespace and enforces capitalization conventions).

The Oddvowel rule serves to avoid confusion between CV and CVV cmapua. A cmapua will never be followed by an odd number of vowels before a consonant or non-letter (except for bizarre situations in acronyms dealt with below). The reason is that the only vowel initial words which might follow a CV or CVV without a break are sequences of VV units; single vowels heading other words are preceded by pauses. The nasty exception is that VCV letterals (something I would like to banish) are preceded directly by other letters in acronyms (and also CVV letterals may be followed by V abbreviations of letterals in acronyms).

```
A0 <- ((([aAEOUeou] !([IAEOUaeiou])) /
(__LWinit 'ha') / (__LWinit 'Ha')) !(Oddvowel))

A <- (__LWinit !(Predicate) (('nu' / 'Nu')
&(('u' / 'no'))))? (('no' / 'No'))? A0 ('noi')? (PA)? !(Oddvowel))

__LWbreak <- (!(Oddvowel)
!((A / ICI / ICA / IGE / I))
 ((', ' !((([ ])* [,])) !((([ ])* Predicate))))?)

A1 <- (A __LWbreak)
```

The class A is inhabited by logical connectives. This is one of the large open-ended classes of structure words handled by the lexer in LIP.

The core A words in class A0 are `a, e, o, u` and the interrogative `ha`. These are never followed by a vowel; that would be a sign of a VV attitudinal.

The full A class includes various trimmings. An A word starting with U or NO may be converted with NU. This is followed by an optional NO. There follows the A0 core. An optional NOI of right negation follows, followed by the very complex possibility of a full dress PA word (preservation of which is open to debate; there are questions about APA words).

The `__LWbreak` class terminates cmapua; it appears where it does because it needs to allude to the A class. It checks the Oddvowel condition, checks

that we are not followed immediately by an A word [or some other vowel initial classes starting with I] (because a comma must intervene), then places an optional comma, which may not be followed by another comma [other terminal punctuation should also be ruled out] or a predicate; this comma is designed to appear as a break between structure words, mandatory before an A, optional in other places, and occasionally used for nontrivial grammatical purposes as in LEPO vs LE,PO.

It might be a good idea to have an actual syllable which can be used as an alternative to a pause to terminate cmapua.

The A1 class is A followed by the LWbreak: this is a fulldress A word rather than a word component.

```
ACI <- (A __LWinit 'ci' __LWbreak)

AGE <- (A __LWinit 'ge' __LWbreak)
```

grammatical variants of A words with different grouping properties to be seen below in the grammar.

```
BI <- (__LWinit ('bia' / 'bie' / 'cie' / 'cio'
/ 'Bia' / 'Bie' / 'Cie' / 'Cio' / 'Bi' / 'bi') __LWbreak)")
```

The predicate BI of identification and other predicates with the same grammar in cmapua form.

```
CA0 <- ((__LWinit 'ca') / 'ce' / 'co' / 'cu'
/ 'ze' / 'Ca' / 'Ce' / 'Co' / 'Cu' / ('Ze' !(Oddvowel)))

CA1 <- (__LWinit ((('nu' / 'Nu') &(('cu' / 'no'))))?
(('no' / 'No'))? CA0 ('noi')? (PA)? !(Oddvowel))

CA <- (__LWinit &(caprule) CA1 __LWbreak)
```

A series of more tightly binding logical connectives (which do not have to be preceded by pauses). As with the PA series, the first rule gives the core words, the second allows attachment of optional extras (the same ones as for the A words) and is a word component rather than a freestanding word, the last is a freestanding CA word.

```
CI <- (__LWinit ('ci' / 'Ci') __LWbreak)

CUI <- (__LWinit ('cui' / 'Cui') __LWbreak)
```

A couple of miscellaneous cmapua, the hyphen CI (which has various uses) and the left mark CUI in metaphors

```
NIO <- (('kua' / 'gie' / 'giu' / 'hie' / 'hiu'
/ 'kue' / 'Kua' / 'Gie' / 'Giu' / 'Hie' / 'Hiu'
/ 'Kue' / 'nea' / 'nio' / 'pea' / 'pio' / 'suu'
/ 'sua' / 'tia' / 'zoa' / 'zoi' / 'Nea' / 'Nio'
/ 'Pea' / 'Pio' / 'Suu' / 'Sua' / 'Tia' / 'Zoa'
/ 'Zoi' / 'ho' / 'ni' / 'ne' / 'to' / 'te' / 'fo'
/ 'fe' / 'vo' / 've' / 'pi' / 'Re' / 'Ru' / 'Sa'
/ 'Se' / 'So' / 'Su' / 'Ho' / 'Ni' / 'Ne' / 'To'
/ 'Te' / 'Fo' / 'Fe' / 'Vo' / 'Ve' / 'Pi' / 're'
/ 'ru' / 'sa' / 'se' / 'so' / 'su' / 'Hi' / 'hi') !(Oddvowel))
```

The atomic quantity words.

```
TAIO <- (((V !(Predicate) !(Name) 'ma')
/ (V !(Predicate) !(Name) 'fi')
/ (C 'ai') / (C 'ei') / (C 'eo')) (!(Oddvowel)
/ &((((V2 [z]))* ((V2 !(letter)) / &(TAIO))))))
```

The basic letterals. I propose eventually to eliminate the VCV letterals: I have ZIV in mind for lower case. I regard the VCV letterals as a really annoying source of phonetic exceptions.

```
NI1 <- (NI0 (('moa' / 'ma'))* !(Oddvowel))

RA <- (('ra' / 'ri' / 'ro' / 'Ra' / 'Ri' / 'Ro') !(Oddvowel))
```

The form moa for 000 has been replaced with MOA to avoid conflict with the pronoun MO. The form ma for 00 may need to be moved if we can expect conflicts with names of capital letters; but I may be able to eliminate the latter. Actually, there may be no possibility of conflict: both forms occur as postfixes to disjoint classes of words.

The class RA is used to control for the fact that final RA, RI, RO in a math string has a quite different meaning (if not solitary) and changes the grammar.

```
NI <- (__LWinit ('ie')? ((((RA / (NI1 &(NI1))))* NI1) / RA)
!((([ ])+ !(Predicate) (NI1 / RA))) (([,] &(([ ])+
!(Predicate) (NI1 / RA)))))? (Acronym2)? ('cu')? !(Oddvowel))
```

This is an open ended class of general quantity words. It begins with the optional IE of interrogative quantification, continues with a solitary RA or a chain of RA and NI1 units ending in a NI1, followed by an option allowing a comma break between quantity words, followed by a possible acronym expressing a dimension, followed by an optional CU. Aspects of this definition call for detailed comment. There might be bugs here; I think in fact that it works but is very subtle.

Breaks in quantity words have serious semantic effects in many situations, and my parser enforces the rule that any break in a quantity word must be marked with an explicit comma standing for a pause in speech. This is more specific that the structure word break in LWbreak.

Query: I don't know of the cu words are really in class NI.

```
L("DA0 <- (__LWinit ('tao' / 'tio'
 / 'tua' / 'Tao' / 'Tio' / 'Tua' / 'mio'
 / 'miu' / 'muo' / 'muu' / 'Mio' / 'Miu'
```

```
 / 'Muo' / 'Muu' / 'toa' / 'toi' / 'too'
/ 'tou' / 'tuo' / 'tuu' / 'suo' / 'Toa'
 / 'Toi' / 'Too' / 'Tou' / 'Tuo' / 'Tuu'
/ 'Suo' / 'Hu' / 'ba' / 'be' / 'bo' / 'bu'
/ 'da' / 'de' / 'di' / 'do' / 'du' / 'Ba'
 / 'Be' / 'Bo' / 'Bu' / 'Da' / 'De' / 'Di'
/ 'Do' / 'Du' / 'mi' / 'tu' / 'Mi' / 'Tu'
/ 'Mu' / 'Ti' / 'Ta' / 'mu' / 'ti' / 'ta'
/ 'hu' / 'mo' / 'Mo') !(Oddvowel))")
```

Atomic pronouns, in the broadest sense.

```
L("Acronym <- (&(caprule) ('mie' / 'Mie') (!(Oddvowel)
/ &(TAI0)) !(NI1) ((NI1 / TAI0 / (([z])? V2 (!(V2)
/ ([z] &(V2))))))+ !((([ ])* !(Predicate)
(NI1 / TAI0 / (([z])? V2 (!(V2) / ([z] &(V2)))))))
(([,] &((([ ])+ !(Predicate) (NI1 / TAI0 /
([z] V2 (!(V2) / ([z] &(V2)))))))))?)")

L("Acronym2 <- (&(caprule) ('mue' / 'Mue')
 (!(Oddvowel) / &(TAI0)) !(NI1)
((NI1 / TAI0 / (([z])? V2 (!(V2) /
([z] &(V2))))))+ !((([ ])* !(Predicate)
(NI1 / TAI0 / (([z])? V2 (!(V2) / ([z] &(V2)))))))
(([,] &((([ ])+ !(Predicate) (NI1 / TAI0 /
([z] V2 (!(V2) / ([z] &(V2)))))))))?)")
```

The front markers with acronyms are a new idea. The first form is for use of an acronym as a preda; the second uses it as a dimension to attach to a number. This is a complex, exception-generating and deeply unsatisfactory feature of the grammar, to my mind. This is partly but not entirely due to the presence of the VCV literals. The front marker to my mind is essential to avoid conflict between acronyms and the much more important appearance of series of letterals as series of arguments.

The fact that acronyms are used in two essentially different ways (as predicates and as dimensions) bothers me (and seems to require two different

front markers). The fact that acronyms are the *only* feature of the grammar that generates a certain phonetic feature (CVV/V junctures in little words) bothers me (and the -eo series cannot participate in such a configuration). I am entirely open to ridding the language of them entirely, as it seems very odd to have such a burden rest on a small part of the language.

Note that the front marker cannot be followed by a VCV letteral; in this context they must be abbreviated. Of course one way to deal with the VCV letterals and the juncture issue would be to require that vowels *always* be abbreviated in acronyms.

```
TAI <-  __LWinit (TAI0/('gao'/'Gao')!V2[ ]*(Name/Predicate
/C V2 V2 (!Oddvowel/&TAI0)/C V2 (!Oddvowel/&TAI0))) __LWbreak
```

This is the class of letteral words usable as variables. An important difference from LIP is that any letteral for me is **just one letter**. Sequences of letter words should be interpreted most often as sequences of distinct letteral argument as in `Mi donsu baicai` which my parser reads as "I give B to C", a reading which LIP will only take if it is forced with a comma and actual pause between BAI and CAI. LIP reads it as "I give BC", reading BC as a single variable, which should not be the primary reading.

Letterals can be subscripted with numerals (see rule DA1).

The use of GAO to create complex letterals is a proposal of John Cowan of which I approve. These letterals may be used as pronouns but may not be subscripted (at least for now).

The reason that I want an initial marker on acronyms is again to prevent any confusion of sequences of letterals in acronyms with the practically more important use of these as sequences of pronouns.

```
Parsedef "DA1<-  __LWinit (TAI0/DA0) ('ci' NI)?!Oddvowel";

Parsedef "DA<- __LWinit DA1 __LWbreak";
```

This is the original class of pronouns. Notice that numeral-subscripted simple letterals are supported; a bug preventing this was working.

```
Parsedef "DIE<- __LWinit('die'/ 'fie'/ 'kae'/ 'nue'/ 'rie'
/'Die'/ 'Fie'/ 'Kae'/ 'Nue'/ 'Rie') __LWbreak";
```