Switchboard SWBD-DAMSL Shallow-Discourse-Function Annotation
Coders Manual, Draft 13

Daniel Jurafsky*, Elizabeth Shriberg+, and Debra Biasca*

University of Colorado at Boulder** & **+SRI International

August 1, 1997

Also available as http://stripe.colorado.edu/~jurafsky/manual.august1.html

1a. Introduction

When we speak about discourse or conversational knowledge, we can talk about a number of different levels. At the level of plans and intentions, we can describe a conversation in terms of the high-level goals and plans of the participants. At the level of focus, we can describe a conversation in terms of center of attentional focus. We might call these intentional or attentional models **deep discourse structure**. At the level of speech acts, we can model the speech act type of each utterance. Or we can model sociolinguistic facts about conversation structure such how participants might expect one type of conversational units to be responsed to by another (adjacency pairs). We refer to these latter two types of discourse structure as **shallow discourse structure**.

This manual describes a completed project which used a shallow discourse tagset of approximately 60 basic tags (plus combinations) to tag 1155 5-minute conversations, comprising 205,000 utterances and 1.4 million words, from the Switchboard corpus of telephone conversations. In particular, this is the thirteenth draft of the instruction manual for the discourse coders of the Discourse Language Model group of the Johns Hopkins WS97 summer large-vocabulary conversational speech recognition (LVCSR) workshop, which includes final statistics now that the coding has now been done.

The main purpose of our label set is to label these Switchboard conversations for training **stochastic discourse grammars** so as to build better Language Models (LM) for Automatic Speech Recognition (ASR) of Switchboard. To that end the label-set incorporates both traditional sociolinguistic and discourse-theoretic rhetorical relations/adjacency-pairs as well as some more-form-based labels. Furthermore, the labelset is structured so as to allow labelers to annotate a Switchboard conversation in about 30 minutes, by editing it with any platform-independent editor (hence the short label-names, and the use of some rich cross-dimension labels). We expect these labeled conversations also to be useful for NLP and Conversational Analysis (CA) research.

The labels were designed to be applied based on the Switchboard *written transcriptions*; this caused the label set to be somewhat more shallow than it could have been with the ability to listen to each utterance. We hope that this shallowness was balanced by the coverage; labeling quickly (conversations took around 30 minutes to label) allowed us to cover much more data.

The labeling project started March 1 1997, and finished July 5, 1997. 8 labelers are CU Boulder linguistics grad students: Debra Biasca (supervisor), Marion Bond, Traci Curl, Anu Erringer, Michelle

Gregory, Lori Heintzelman, Taimi Metzler, Amma Oduro. 1155 conversations were labeled; the average one has 144-turns, 271 utterances. By the end of the labeling the labelers took about a half hour to label a conversation (conversations averaged 5 minutes). We are currently using the Kappa statistic (Carletta 1996, Carletta et al (in press)) to assess labeling accuracy; average pairwise Kappa (as of the end of the project) was **.80**. The Discourse Language Modeling research group includes Becky Bates, Noah Coccaro, Thomas Crystal, Carol van Ess-Dykema, Dan Jurafsky, Rachel Martin, Marie Meteer, Klaus Ries, Liz Shriberg, Andreas Stolcke, and Paul Taylor, and external advisors who gave extremely helpful comments on the tagset were James Allen, Barbara Fox, Julia Hirschberg, Susann LuperFoy, Marilyn Walker, and Nigel Ward.

The current version of the discourse tag-set is designed as an augmentation to the Discourse Annotation and Markup System of Labeling (DAMSL) tag-set. For that reason it is designed to be read together with "James Allen and Mark Core. 1997. Draft of DAMSL: Dialog Act Markup in Several Layers. March 21, 1997", which gives the theoretical background of DAMSL-style tagging, and with Meteer (1995) "Dysfluency Annotation Stylebook for the Switchboard Corpus", which gives the annotation instructions for the previous years' annotation of SWBD with slash units.

There is a deterministic mapping between about 80% of the "SWBD-DAMSL" labels in this document and the standard DAMSL labels, (except that some of the SWBD-DAMSL labels further subdivide the DAMSL labels). In a few cases a mapping is not possible, usually for one of two reasons: either we and the coders were unable to accurately mark a distinction which the March 21 1997 DAMSL standard requires (for example the distinction between Assert and Reassert), or we felt the need to mark extra distinctions which DAMSL doesn't require. However in a few other cases we have proposed a minor augmentation to DAMSL which is not simply "added-subtypes"; one such example is modifying Self-Talk to include not one but 2 kinds of non-second-person- directed talk; self-talk and third-party talk). We have not attempted in this Coder's Manual to map these DAMSL-style tags into other theories of speech acts, intention-tracking in discourse, conversational analysis, discourse commitment, centering, etc. See the DAMSL standard for more theoretical justifications for the particular tagging philosophy.

In addition to this set of labels, the WS97 project has marked other acoustic features (f0, energy, speaking rate, snr etc) of each utterance in Switchboard in another, distinct database. In addition, some of the utterances will have hand-marked pitch-accent labels and phonetic transcriptions.

1b. SWBD-DAMSL and the WS97 Language Modeling Project

The main goal of the summer Johns Hopkins LVSCR Workshop-97 summer project (July 14 - Aug 22, 1997) is to use discourse information to improve the Language Model (LM) on the Switchboard (SWBD) task. We clustered the 220 tags into 42 clustered tags, and then trained separate trigram LMs from the utterances in each of the 42 classes. Our goal is then to build a number of different 'Utterance-Type detectors', based on different sources of evidence for Utterance-type: prosodic, acoustic, lexical, and discourse sequence. Given an utterance from the test-set, we will use the predicted utterance-type to select the appropriate utterance-type-specific language model for the utterance. We can summarize this research plan as follows:

- Code: Hand-label utterances with discourse-tags and pitch information
- Cheat: Check: Would perfect utterance-type knowledge improve LM?
- Detect: Train various automatic utterance-type detectors

- Improve: Use utterance-type to improve SWBD LMs

We will explore various algorithms for utterance-type detection, and various combinations of them. These will include:

- Pitch-Accent: HMM-based prosodic algorithm (cf. Taylor et al 1996, Takagi and Itahashi 1994, Terry et al 1994, Waibel 1988)
- Other Acoustics: Frame based analysis of raw F0, duration, energy, SNR, speaking rate
- Discourse Grammar: "Discourse-level" HMM or N-gram to predict next-utterance-type (Kita et al 1996, Reithinger and Maier 1995, Reithinger et al 1996, Sarukkai and Ballard 199 6, Suhm and Waibel 1994, Woszczyna and Waibel 1994, Yamaoka, and Iida 1990}
- Cue Phrases/POS/Words: Using word or tag stream to predict utterance type (Garner et al 1996, Hirschberg and Litman 1993)

1c. The 42 Clustered SWBD-DAMSL Labels

There were 220 tags used in the coding; 130 of these occurred less than 10 times each, so for our initial experiments we clustered the 220 tags into 42 larger classes. We did the clustering by removing the secondary carat-dimensions (^2,^g,^m,^r,^e,^q,^d), with 5 exceptions. The exceptions: we left qy^d (Declarative yes-no Questions) , qw^d (Declarative wh-questions) and b^m (Signal-Understanding-via-Mimic), and we folded the few examples of nn^e into ng, and ny^e into na. Then, we grouped together some tags that had very little training data; those tags that appear in the following list were grouped with other tags on the same line.

```
qr qy
fe ba
oo co cc
fx sv
fo o fw " by bc
aap am
arp nd
```

We also removed any line with a "@" (since @ marked slash-units with bad segmentation).

Here are the resulting 42 classes with their final counts in the WS97 training set (out of 197,489 training-set utterances, 1.4M words, 1115 conversations); (the remaining 40 conversations were saved for the test sets and so we do not include them in the statistics).

SWBD-DAMSL	SWBD	Example	Cnt	%
Statement-non-opinion	sd	*Me, I'm in the legal department.*	72,824	36%
Acknowledge (Backchannel)	b	*Uh-huh.*	37,096	19%
Statement-opinion	sv	*I think it's great*	25,197	13%
Agree/Accept	aa	*That's exactly it.*	10,820	5%
Abandoned or Turn-Exit	% -	*So, -*	10,569	5\%
Appreciation	ba	*I can imagine.*	4,633	2%
Yes-No-Question	qy	*Do you have to have any special training?*	4,624	2%

Non-verbal	x	*[Laughter], [Throat_clearing]*	3,548	2%
Yes answers	ny	*Yes.*	2,934	1%
Conventional-closing	fc	*Well, it's been nice talking to you.*	2,486	1%
Uninterpretable	%	*But, uh, yeah*	2,158	1\%
Wh-Question	qw	*Well, how old are you?*	1,911	1%
No answers	nn	*No.*	1,340	1%
Response Acknowledgement	bk	*Oh, okay.*	1,277	1%
Hedge	h	*I don't know if I'm making any sense or not.*	1,182	1%
Declarative Yes-No-Question	qy^d	*So you can afford to get a house?*	1,174	1%
Other	o,fo,bc,by,fw	*Well give me a break, you know.*	1,074	1%
Backchannel in question form	bh	*Is that right?*	1,019	1%
Quotation	^q	*You can't be pregnant and have cats*	934	.5%
Summarize/reformulate	bf	*Oh, you mean you switched schools for the kids.*	919	.5%
Affirmative non-yes answers	na,ny^e	*It is.*	836	.4%
Action-directive	ad	*Why don't you go first*	719	.4%
Collaborative Completion	^2	*Who aren't contributing.*	699	.4%
Repeat-phrase	b^m	*Oh, fajitas*	660	.3%
Open-Question	qo	*How about you?*	632	.3%
Rhetorical-Questions	qh	*Who would steal a newspaper?*	557	.2%
Hold before answer/agreement	^h	*I'm drawing a blank.*	540	.3%
Reject	ar	*Well, no*	338	.2%
Negative non-no answers	ng,nn^e	*Uh, not a whole lot.*	292	.1%
Signal-non-understanding	br	*Excuse me?*	288	.1%
Other answers	no	*I don't know*	279	.1%
Conventional-opening	fp	*How are you?*	220	.1%
Or-Clause	qrr	*or is it more of a company?*	207	.1%
Dispreferred answers	arp,nd	*Well, not so much that.*	205	.1%
3rd-party-talk	t3	*My goodness, Diane, get down from there.*	115	.1%
Offers, Options Commits	oo,cc,co	*I'll have to check that out*	109	.1%
Self-talk	t1	*What's the word I'm looking for*	102	.1%
Downplayer	bd	*That's all right.*	100	.1%

Maybe/Accept-part	aap/am	*Something like that*	98	<.1%
Tag-Question	^g	*Right?*	93	<.1%
Declarative Wh-Question	qw^d	*You are what kind of buff?*	80	<.1%
Apology	fa	*I'm sorry.*	76	<.1%
Thanking	ft	*Hey thanks a lot*	67	<.1%

1d. The Entire Label set and its mapping to DAMSL tags

Mapping of WS97 tags to DAMSL tags (see Allen and Core March 21 1997)

Bold-faced codes are new SWBD-DAMSL codes not in DAMSL.

DAMSL	SWBD
Communicative-Status	
Uninterpretable	% with no a final "-/"
Non-verbal	laughter, coughs, etc)
Abandoned	% together with -\/
Self-talk	t1
3rd-party-talk	t3
Information-level	
Task	DEFAULT
Task-management	^t
Communication-management	^c (but ^c is only a subpart of Comm-management)
Other	NOT CURRENTLY MARKED
Forward-Communicative-Function	
Statement	s
Assert	(not marked)
Reassert	(not marked)
Statement-non-opinion	sd
Statement-opinion	sv

Influencing-addressee-fut-actn	
Open-option	oo
Directive	
Info-request	qy, qw, qo, qr, qrr, ^d, ^g
Yes-No-question	qy
Wh-Question	qw
Open-Question	qo
Or-Question	qr
Or-Clause	qrr
Declarative-Question	^d
Tag-Question	^g
Action-directive	ad
Committing-speaker-future-action	
Offer	co
Commit	cc
Other-forward-function	
Conventional-opening	fp
Conventional-closing	fc
Explicit-performative	fx
Exclamation	fe
Other-forward-function	fo
Thanking	ft
You're-Welcome	fw
Apology	fa
Backwards-Communicative-Function	
Agreement	
Accept	aa
Accept-part	aap
Maybe	am
Reject-part	arp
Reject	ar
Hold before answer/agreement	^h

Understanding	
Signal-non-understanding	br, br^m
Signal-understanding	
Acknowledge	b,bh
Acknowledge-answer	bk
Repeat-phrase	^m
Completion	^2
Summarize/reformulate	bf
Appreciation	ba
Sympathy	by
Downplayer	bd
Correct-misspeaking	bc
Answer	DEFAULT-for-qw,ny,nn,na,nd,ng,no,sd^e,sv^e,^h
Yes answers	ny
No answers	nn
Affirmative non-yes answers	na
Negative non-no answers	ng
Other answers	no
No plus expansion	nn^e
Yes plus expansion	ny^e
Statement expanding y/n answer	sd^e,sv^e
Expansions of y/n answers	^e
Dispreferred answers	nd
Other	
Information-relation	NOT CODED
Quoted material	^q
Hedge	h
Segment (multi-utterance)	+
Double labels	x;y, [where x is the preferred label]
Transcription errors: slash units	o@, [anycode]@, +@
Transcription errors: typographical errors	*

Alphabetic listing of tags

```
(useful mnemonics:

  q      Question
  s      Statement
  b      Backchannel/Backwards-Looking
  f      Forward-Looking
  a      Agreements
  %  indeterminate, interrupted, or contains just a floor holder (see manual)
  (^u  [on anything] unrelated response (first utt is NOT response to previous q)
  *  comment  (followed by "*[[comment...]]" after transcription to explain)
  +  continued from previous by same speaker
  @,o@,+@  incorrect transcription (can add comment to specify problem further)
  ^2 collaborative completion
  ^c  about-communication
  ^d  declarative question (question asked like a structural statement)
  ^e  [on statements] elaborated reply to y/n question
  ^g  tag question (question asked like a structural statement with a question tag a
  ^h  hold (often but not always after a question) ('let me think'; question in resp
  ^m  mimic other
  ^q  quotation
  ^r  repeat self
  ^t  about-task
  aap Accept-part
  ad Action-directive  "Go ahead", "We could go back to television shows"
  aa Accept         "ok" , "i agree"
  am Maybe
  ar Reject "no",
  arp Reject-part
  b default agreement or continuer (uh-huh, right, yeah)
  b^m  Repeat-phrase
  ba assessment/appreciation ("I can imagine")
  bc Correct-misspeaking
  bd Downplaying-reponse-to-sympathy/compliments ("That's all right","that happens")
  bf reFormulate/summarize; paraphrase/summary of other's utterance (as opposed to a
  bh rhetorical question continuer ("Oh really?")
  bk ACKNOWLEDGE-ANSWER    "Oh, okay"
  br Signal-non-understanding (request for repeat)
  br^m Signal-non-understanding via mimic
  br^c non-understanding due to problems with phone line
  by sYmpathetic comment ("I'm sorry to hear about that")
  cc Commit
  co Offer
  fa Apology "Apologies" (this is not the "I'm sorry" of sympathy which is "by")
  fc Conventional-closing
  fe Exclamation "Ouch"
  fo Other-forward-function
  fp Conventional-opening
  ft Thanks "Thank you"
  fw Welcome "You're welcome"
  fx Explicit-performative  ("you're filed" )
  na a descriptive/narrative statement which acts as an affirmative answer to a ques
  nd aNswer Dispreferred (Well...)
  ng a descriptive/narrative statement which acts as a negative answer to a question
  nn  no or variations (only)
  no a response to a question that is neither affirmative nor negative (often "I don
  ny  yes or variations (only)
  o other
  oo Open-option  "We could have lamb or chicken"
  qh  rhetorical question
  qo  open ended question
  qr  alternative ('or') question
```

```
  qrr an or-question clause tacked onto a yes-no question
  qw  wh-question
  qy  yes/no question
  sd  descriptive and/or narrative (listener has no basis to dispute)
  sv  viewpoint, from personal opinions to proposed general facts  (listener could h
  t1  self-talk
  t3  3rd-party-talk
  x   nonspeech
```

Finally, for reference, here are the original 226 tags:

```
70495 sd
36251 b
25709 sv
17798 +
15590 %
10159 aa
4531 ba
3787 qy
3693 x
2833 ny
2406 fc
2102 b^r
1940 sd^e
1893 qw
1343 sd(^q)
1257 bk
1233 nn
1221 qy^d
1218 h
1044 bh
 976 ^q
 940 bf
 932 sd^t
 916 aa^r
 808 o
 765 na
 720 ^2
 688 b^m
 666 ad
 644 qo
 563 qh
 556 ^h
 440 qy^g
 303 ar
 302 sv(^q)
 291 ng
 279 no
 248 sd^r
 238 br
 219 qr
 207 fp
 198 qrr
 196 ny^r
 181 nd
 157 sv^t
 137 nn^r
 134 fe
 131 fc^m
 118 sv^e
 117 t3
```

```
114 qy^t
103 ba^r
102 t1
 96 bd
 92 ^g
 88 sv^r
 80 qw^d
 76 ft
 76 fa
 69 aa^m
 67 sd^m
 64 ad^t
 59 br^m
 57 aap
 50 sd^c
 49 qw^t
 49 co
 44 am
 41 ar^r
 37 sd
 37 na^r
 35 cc
 34 na^m
 30 bk^r
 29 qy^r
 29 fc^t
 29 "
 25 sv^m
 23 arp
 22 sd(^q)^t
 21 qy^h
 21 bk^m
 19 sv
 19 qy^g^t
 19 by
 18 fc^r
 16 qy^m
 16 qy^c
 15 fp^m
 14 qy^d^t
 14 qw^r
 13 qr^d
 13 co^t
 11 qw^h
 11 bc
 10 sd^e^t
  9 na^t
  9 fx
  7 qy^2
  7 ny^m
  7 bd^r
  6 qy^d^r
  6 qrr^t
  6 qo^t
  6 nn^m
  6 bh^m
  6 bf^r
  6 ad(^q)
  6 ^q^t
  5 sd^e^r
  5 sd^e^m
```

```
5 sd^2
5 qrr^d
5 nn^e
5 fo
5 ^2^g
4 qy^d^m
4 qy(^q)
4 qo^d
4 qh^m
4 oo
4 o^r
4 no^t
4 ng^r
4 h^r
4 fw
4 ad^r
4 ad^c
3 sv^c
3 sv^2
3 qy
3 qw^g
3 qw^d^t
3 qr^t
3 nd^t
3 fp^r
3 co^c
3 bh^r
3 bf^m
3 ba^m
3 b^m^t
3 aa^t
3 aa^2
2 qy^g^r
2 qy^g^c
2 qy^d^h
2 qy^c^r
2 qw^m
2 qw^c
2 qw
2 qh^r
2 qh^h
2 oo^t
2 o^t
2 ny^e
2 ny^c
2 no^r
2 ng^m
2 h^t
2 fa^c
2 cc^r
2 br^r
2 bf^t
2 bf^g
2 bf(^q)
2 bc^r
2 b^m^r
2 b^m^g
2 am^r
2 ad
2 ^q^r
2 ^h^r
```

```
1 t1^t
1 sv^e^r
1 sv;sd
1 sd^e(^q)^r
1 sd;sv
1 sd;qy^d
1 sd;no
1 sd,sv
1 sd,qy^g
1 sd(^q)^r
1 qy^d^c
1 qy^d(^q)
1 qw^r^t
1 qw^d^c
1 qw(^q)
1 qr(^q)
1 qo^r
1 qo^d^c
1 qh^g
1 qh^c
1 qh(^q)
1 qh
1 oo(^q)
1 o^c
1 ny^t
1 ny^c^r
1 nn^t
1 nn^r^t
1 ng^t
1 na^m^t
1 h^m
1 h,sd
1 ft^t
1 ft^m
1 fa^t
1 fa^r
1 cc^t
1 bk^t
1 bf^2
1 bf
1 ba,fe
1 b^t
1 b^2
1 ar^m
1 aap^r
1 aap^m
1 aa^h
1 aa,ar
1 ^m
1 ^h^t
1 ^2^t
1 ^2^r
1 +,ny
```

1e. A Sample (Short) Conversation

FILENAME: 4360_1599_1589

^h	A.1 utt1: {F Uh, } let’s see. /
%	A.1 utt2: How [about, + {F uh, } let’s see, about] ten years ago, /
qo	A.1 utt3: {F uh, } what do you think was different ten years ago from now? /
sv	B.2 utt1: {D Well, } I would say as, far as social changes go, {F uh, } I think families were more together. /
sv	B.2 utt2: [They, + they] did more things together. /
b	@A.3 utt1: Uh-huh <>. /
sv	B.4 utt1: {F Uh, } they ate dinner at the table together. /
sv	B.4 utt2: {F Uh, } the parents usually took out [time, + {F uh, } {D you know, } more time] than they do now to come with the children and just spend the day doing a family activity. /
b	A.5 utt1: Uh-huh. /
sv	B.6 utt1: {F Uh, } although I’m not a mother, [I, + I] still think that, {F uh, } a lot has changed since ten years ago. /
qo	B.6 utt2: {F Uh, } what # do you # --
%	A.7 utt1: # We, # -/
+	B.8 utt1: -- think about that? /
sv	A.9 utt1: {D Well, } {F uh, } {D actually } ten years from today seems rather short. /
b	B.10 utt1: Yeah. /
sv	A.11 utt1: {F Uh, } {C but } I do agree that, {F uh, } generally [it’s, + society] has sort of, {F uh, } let’s see, rushed everything ahead. /
b	B.12 utt1: Uh-huh. /
h	A.13 utt1: {C And, } {F uh, } I don’t know, /
sv	A.13 utt2: it [leaves, + leaves] a lot of time out for family and things like that. /
sv	A.13 utt3: In other words, they just prioritize their lives differently. /
sv	A.13 utt4: {C But } I think that has a lot to do with economic situation. /
aa	B.14 utt1: Yes. /
qo	B.14 utt2: What about {D like } as far as, {F uh, } social changes in the individual? /
qy	B.14 utt3: # Do # --
%	A.15 utt1: # {F Uh, } # /
+	B.16 utt1: -- you think that the individual has as much time as they did, let’s say, ten, twenty years ago? /
h	A.17 utt1: {F Um. } It depends. /
sv	A.17 utt2: {F Uh, } it’s hard to say because I think people were busy ten twenty years ago too. /
b	B.18 utt1: Uh-huh. /

%	A.19 utt1: {F Uh, } I just , -/
qw	B.20 utt1: {D Well, } [how, + how] old are you? /
sd	A.21 utt1: I’m twenty-eight. /
b^m	B.22 utt1: Twenty-eight. /
bk	B.22 utt2: Okay, /
sd	B.22 utt3: I’m twenty-three. /
b	A.23 utt1: Yeah. /
sd	B.24 utt1: {C So } there’s maybe a five year gap between us. /
b	A.25 utt1: Yeah. /
%	B.26 utt1: {D So, } {F uh. } -/
sv	A.27 utt1: [I just, + I] think that things [[were a bit, + were,] + have been] busy all along. /
sv	A.27 utt2: It’s # just # --
%	B.28 utt1: # {F Huh } # <>. /
+	A.29 utt1: -- a matter where priorities are, [at +] placed.
aa	B.30 utt1: Yes. /
+	A.31 utt1: And that, {F uh, } usually as far as families are concerned, there used to be just one person working and usually the other parent was home. /
b	B.32 utt1: Uh-huh. /
sv	A.33 utt1: {C And } now, {F uh, } it’s pretty much an economic necessity [[of, + for most,] + in most] places for both parents to work. /
qy	B.34 utt1: Do you think it’s an economic [c-, + necessity] /
qrr	B.34 utt2: {C or } do you think that [we’re, + we’re,] {F uh, } all trying to keep up with a certain standard of living? /
sv	A.35 utt1: I think that’s part of it too. /
sv	A.35 utt2: {C But } I do think, -/
qy	B.36 utt1: {E I mean } do you think,
x	A.37 utt1: .
+	B.38 utt1: people really need two cars and --
nn	A.39 utt1: No, /
nn^r	A.39 utt2: no. /
sd^e	A.39 utt3: # I don’t. # /
+	B.40 utt1: -- # a house # in the suburbs {C or, } -/

nn	A.41 utt1: No, /
sd^e	A.41 utt2: I don’t think that. /
sv	A.41 utt3: {C But then } there are a lot of people [that, + that] don’t have that.
b	B.42 utt1: Uh-huh. /
+	A.43 utt1: But, that really do need to work. /
b	B.44 utt1: Uh-huh. /
sv	A.45 utt1: I think maybe those people that really do need to work, both parents, just to survive. - /
sv	A.45 utt2: # {C And # --
b	B.46 utt1: # Yeah. # /
+	A.47 utt1: -- then } there, [th-, +] [is, + is] that other group # that is # --
b	B.48 utt1: # Uh-huh. # /
+	A.49 utt1: -- working to maintain a standard of living --
bk	B.50 utt1: Okay. /
+	A.51 utt1: -- that, {F uh, } they think [is, + is] surviving which is really more luxuries. /
b	B.52 utt1: Uh-huh. /
sv	A.53 utt1: {F Uh, } {C but } [I + I] tend to think that it’s less those people that have the two cars and everything than it is the group that is just trying to survive. /
qy^d	B.54 utt1: [Yo-, + {C so } you] think it’s, - /
qw	B.54 utt2: which group are you saying # is the one trying? # /
sv	A.55 utt1: # I’m saying that # [the, + {F uh, } the] group that is just trying to survive from day to day, where both parents are working --
b	B.56 utt1: Uh-huh. /
+	A.57 utt1: -- is more of the majority [than the, + than the] people that have the higher standard of living. /
sv	A.57 utt2: {C Because } if you look at economics across this country and statistics on who has the money and who the decreasing, {F uh, } middle class in this country --
b	B.58 utt1: Uh-huh. /
+	A.59 utt1: -- I think that that’s, in my opinion, the case. /
bk	B.60 utt1: Okay. /
%	A.61 utt1: {D So. } - /
sd	A.61 utt2: {E I mean } I have met people [[that, + {F uh, } both that,] + that] just want to maintain [a, + the] standard of living and those [that, + that] really need the job. /

b	B.62 utt1: Okay. /
sd	B.62 utt2: {C And then, } sometimes [I, + I] often, {F uh, } find that maybe there's so many different things available to us. [Yo-, +] a microwave, a V C R, a answering machine --
b	A.63 utt1: Uh-huh. /
+	B.64 utt1: -- [[a, + {D you know, } a special,] + a] dishwasher, {F uh, } a refrigerator and some of those items, {F um, } [for the, + for the,] {F uh, } - /
sv	B.64 utt2: {D well } I guess we're sticking more to social changes /
sv	B.64 utt3: {C but, } {F uh } --
b	A.65 utt1: Uh-huh. /
+	B.66 utt1: -- people want all of that /
sv	B.66 utt2: {C and } not all of those are necessities. /
b	A.67 utt1: Right . /
sv	B.68 utt1: {C So } they're trying to, - /
sv	B.68 utt2: it has become a necessity . /

2. Units to label

We are labeling each "slash unit", which is something like a TCU (Sacks, Schegloff and Jefferson 1974). See the Meteer (1995) "Dysfluency Annotation Stylebook for the Switchboard Corpus" for the definition of slash units, and in particular for the heuristics used by the LDC to break complex sentences into slash units. This was done in 1995-1996; for a number of logistical reasons, in this labeling project we are treating these boundaries as unchangeable. In a future version of this document we hope to discuss the differences between these units, TCUs, and the segmentation algorithms to be written up by the DRI.

We will not be fixing what we consider mis-transcriptions, although we will be marking them to be fixed at some future date. As coded originally, the start of a slash unit is either the first word by a speaker in a conversation, or the first word after a previous "/" or "-/"; the end of a slash unit is either "/" or "-/".

A slash unit can consist of exactly one turn, less than one turn, or more than one turn. To determine if a turn is the end of a slash unit:

```
ignore the " -- " and " - " from original transcriptions
"/"      = end of complete unit
"-/"     = end of cut-off unit
Neither  = unit continues to next turn by same speaker
```

To label slash units spanning more than one turn:

- Read up to the end of the slash unit to determine label.
- Place label at the onset of slash unit.
- Use "+" for all later turns within that slash unit.

We mark two kinds of errors in the transcriptions. Segmentation errors (either a slash unit that is too long or too short) are marked by placing an "@" after the discourse tag. Transcription errors (typos, obvious mistranscriptions) are marked with a "*" after the discourse tag.

Both kinds of errors may also have a comment at the end of the line, starting with "*[[" and ending with "]]".

---

3. Communicative-Status

```
Communicative-Status
     Uninterpretable                      % without a final "-/"
     Non-verbal                           x for non-verbal stuff (pure laughter, c
     Abandoned                            % together with -/
     Self-Talk                            t1
     3rd-person-talk                      t3
```

The DAMSL tagset is organized into orthogonal dimensions; every utterance can take a value on each of 5 dimensions. SWBD-DAMSL, by contrast, has fewer dimensions, and Communicative Status is not one of them. In DAMSL an utterance is tagged for Communicative-Status and also the other 4 dimensions, but in SWBD-DAMSL we don't mark any other dimensions on an utterance which has any of the Communicative-Status tags (here for purely practical reasons: we were unable to do it accurately). These utterances could be viewed theoretically as "Underspecified" for the other 4 dimensions.

The DAMSL Abandoned category is marked by adding the "%" tag to those utterances that already end with a "-/". (i.e. abandonment was often already marked by the LDC).

The DAMSL Uninterpretable category has two SWBD-DAMSL subtypes, depending on whether the uninterpretable utterance was verbal or nonverbal. (this distinction is mainly motivated for speech-recognition purposes).

```
  1) A % on an utterance (which doesn't end in "-/") marks uninterpretable
utterances that have verbal material.
  2) x is used for uninterpretable utterances with solely non-verbal material.
```

3.01 %

The % is used if the utterance is cut off in such a way that you can't readily tell what it would have been. A.27 utt 2, below, is not a %, because you could probably figure out that it's an sv:

A.27 utt2: {C but, } {F uh, } I think drug testing, - /

When in doubt, use %. In general, if the utterance has four or fewer words, it is probably '%'. In B.22

utt1, there is sufficient information to tell that an opinion (sv) is being formulated. In B.22 utt2, however, there is insufficient information:

```
sv        B.22 utt1: [ That's, + {F uh, } that's ] a little bit too,{F uh, } - /
%         B.22 utt2: ((it's such)) - /
sv        B.22 utt3: they're trying to make it too much of a crossover thing,  /
qy        B.22 utt4: you know what I mean? /
```

% is also used to mark short "turn exits" (i.e. "Yeah" or "So" or "Or).

3.02 Self and Other-talk (t1 and t3)

Where DAMSL has a "Self-Talk" category, SWBD-DAMSL proposes that this be replaced with the NON-2ND-PERSON-TALK category, which covers all type of talk not-directed at the conversation partner. It would have subtypes "Self-Talk" (labeled "t1") and "3rd-party-talk" (labeled "t3"). 3rd-party-talk is intended to handle talk to other people than the conversation participants, in situations like the following:

```
           B.16 utt4: Could I ask you to hold one minute? /  *[[this is really a Pr

           A.17 utt1:  Uh-huh. /

           B.18 utt1:  I'll be right back.  / *[[ what are these?]]
           B.18 utt2:  # Excuse me, #

%         A.19 utt1:  #  (( Had-, ))  # -/

+         B.20 utt1:  just a moment.  /
sd         B.20 utt2:  They're going to get mad . /

t3         A.21 utt1: <> She had another call.  /
t3         A.21 utt2: <> She has (( just )) three kids, eleven, nine, and eight. /
```

Coder's Heuristic for Self-talk 't1'

If the content of speaker's utterance does not seem to be intended for the listener to respond to, it is 't1' In the example below, the speaker seems to be talking to him/herself. The preceding context of the conversation makes it clear that this question (A.145 utt2) is not being addressed to Speaker B.

```
sd         B.144 utt1: I'll have to tune in. /

sd         A.145 utt1: It's on E S P N, {F uh, }  /
t1         A.145 utt2: at what time,  /
sd         A.145 utt3: I can't remember what time.  /
%         A.145 utt4: It's, {F uh, } {D you know, } - /
sd         A.145 utt5: I can't remember offhand what time. /
```

Things that seem somewhat self-directed like "Hmmm, let's see" or "what else", we are **not** coding as **t1** but rather as **^h** ("hold's").

---

4. Information Level

The SWBD-DAMSL "Informational Level" Dimension is a true dimension like the DAMSL Information Level dimension. The ^t and ^c labels can be added to any other labels from other dimensions.

```
  Information-level
       Task                                 DEFAULT
       Task-management                      ^t
       Communication-management             ^c (but ^c is only a subpart of Comm-man
       Other                                NOT CURRENTLY MARKED
```

4.1 Task-Management ^t

---

^t means "task", and is used on utterances which constitute task-management. The "task" of SWBD is hereby defined as "having and recording a conversation within X minutes about some topic area Y".

```
sv^t    A.1 utt1:  {F Uh, } the question was kind of interesting to
sv^t    A.45 utt1:  {F Uh, } probably need to try to get back on the topic
sv^t    A.1 utt2:  I think the first thing they said, - /
sd^t    A.21 utt3:  Third question was how [ m-, + ]  (( ))  serving
              for their own gains do you think goes on, - /
___________
sd^t          A.1 utt2: I almost forgot what the topic  was. /

b          B.2 utt1:  Okay.  /
%          B.2 utt2: {F Uh, } # based, # -/

sd^t          A.3 utt1:  # {F Uh, } # {C but } I know what it is. /
```

4.2 Communication-Management ^c (and fp, fc, b, b^m, see below)

The SWBD-DAMSL **^c** tag is an orthogonal dimension which is used to mark communication problems or specific remarks adressing communication:

```
qw^c          A.96 utt1:  Pardon me? /

qy^d^c          A.5 utt1:  I heard a laugh in the background. /

sd^c           A.44 utt1:  I think a train went by. /
```

```
sd^c            B.2 utt2:  I couldn't hear you? /
```

The SWBD-DAMSL **^c** tag is only a subset of the DAMSL Communication-Management tag. Communication-Management includes a number of other things which SWBD-DAMSL does not code with **^c** Following is a paragraph from Allen and Core (page 6), split out on separate lines together with the SWBD-DAMSL tag which corresponds with each SWBD function:

```
"Utterances at this level include conventional phrases that maintain contact,
perception, and understanding during the communication process, and include

fp    greetings (perFormative--oPening) (e.g., "hello"),
fc    closings ("Good Bye"),
b     acknowledgements (e.g., "Okay", "uh-huh",
b^m   or repeating parts of what the speaker said),
^h    stalling for time, (e.g., "Okay", "Let me see"),
??    or signals of speech repair (e.g. "oops") or misunderstandings."
^c    They also might address the communication process explicitly, say to establish
^c    the communication channel (e.g. "Are you there?", and answering with "I'm here
br,^c    to address communication problems (e.g. "Can't hear you; there's static on
      or to explicitly manage delays or maintain the turn (e.g "Wait a minute").
```

So when mapping from SWBD-DAMSL to DAMSL, the tags **fp, fc, b, ^h,**, and **br** can be mapped automatically to Communication-Management.

5. Forward-Communicative-Function

The mapping between SWBD-DAMSL and DAMSL is most complex in the Forward-Communicative-Function and Backwards-Communicative-Functions. In DAMSL, these are completely orthogonal, allowing for 13 (Forward) x 12 (backwards) or 156 possible Forward-Backward combinations. In SWBD-DAMSL, while all these 156 combinations are still technically open to the labeller, we have created "shortcut" codes for common combinations of forward and backward function.

For the first 200 conversations we also allowed the labelers to code any combination of Forward and Backwards function (with the goal of searching for extra combinations); we then took these combinations and made standard labels of them; there were very few.

DAMSL	SWBD
Forward-Communicative-Function	
Statement	s
Assert	(not marked)
Reassert	(not marked)
Statement-non-opinion	sd
Statement-opinion	sv
Influencing-addressee-fut-actn	
Open-option	oo
Directive	
Info-request	qy, qw, qo, qr, qrr, ^d, ^g

Yes-No-question	qy
Wh-Question	qw
Open-Question	qo
Or-Question	qr
Or-Clause	qrr
Declarative-Question	^d
Tag-Question	^g
Action-directive	ad
Committing-speaker-future-action	
Offer	co
Commit	cc
Other-forward-function	
Conventional-opening	fp
Conventional-closing	fc
Explicit-performative	fx
Exclamation	fe
Other-forward-function	fo
Thanking	ft
You're-Welcome	fw
Apology	fa

5.1 Statements "s"

Statements are the most common label in SWBD-DAMSL, comprising 45% of the tokens. One of the SWBD-DAMSL/DAMSL mapping difficulties occurs with statements. SWBD-DAMSL statements are not differentiated into DAMSL's "Assert", "Reassert" and "Other Statement". This is not for theoretical reasons; it was just not possible for us to distinguish a "Reassert" from an "Assert" in casual conversation. (In task-oriented dialog, the task often imposes enough structure on the organization and content of the conversation (Grosz 1978) that it is possible to say absolutely if some piece of information concerning the task has been previously transmitted; we were unable to do this in casual conversation).

As a result we have mapped all SWBD-DAMSL labels starting with "s" into the more abstract "Statement" node of the DAMSL hierarchy, rather than the more specific "Assert", "Reassert" or "Other Statement".

5.1.1 sd and sv

SWBD-DAMSL makes another pragmatic distinction not made in DAMSL, the distinction between "descriptive/narrative/personal" statements (sd) and "other-directed opinion statements" (sv). The distinction was designed to capture the different kinds of responses we saw to opinions (which are often countered or disagreed with via further opinions) and to statements (which more often get continuers/backchannels).

We have not yet decided whether this sd/sv distinction has been fruitful. We trained separate trigram language models on the two sets, and they looked somewhat distinct. But the distinction was very hard to make by labelers, and accounted for a large proportion of our interlabeler error.

We would just list "sd" and "sv" as subtypes of "Assert" except that they technically are an orthogonal dimension from the new/old "Assert"/"Reassert" distinction.

Coders Heuristics

When in doubt, it is probably sd.

Use sd when speaker is telling a story and the topic is personal (i.e., look for "I" "we" referring to speaker and his/her family or other acquaintances, not "we" referring to speaker and listener, statement about her dog, her house, her neighborhood, etc, or a statement where speaker voices his/her opinion about that topic. If it helps, think of these as ’personal statements.’ one way to think about this is that sd used to have 3 subtypes:

> narrative (pieces of story)
> declarative statements (boulder is north of denver)
> personal statements (I was born in chicago, I get along well with my boss)

The third one of these looks like those "sv" opinions, but isn’t, because it’s something the listener doesn’t really "get to be an expert on". If the statement is about something more general, that the listener could conceivably have their own (possibly differing) opinion about, then it will be sv.

Examples of sd, where speaker A is talking about his cat, from conv. sw01_4019:

```
qw           B.8 utt1:  How about you? /
sd           A.9 utt1:  {D Well, } we have a cat, {F um, }  /
sd           A.9 utt2: he’s probably, {F oh, } a  good two years old, big,
                       old, fat and sassy tabby. /
. . .
 b          B.20 utt1:  {F Huh. } /
 +          A.21 utt1:  -- some reason.  /
sd          A.21 utt2:  He’s, {F uh, } been so mean to her . /
. . .

%           A.29 utt4: # {C so. } # -/
b           B.30 utt1:  # Uh-huh. # /
```

```
sd          A.31 utt1:  {C But } he’s a very possessive cat.  /
```

Example of sd, where speaker A is talking about raising boars and pigs, something he is ’expert’ on according to the conversation:

```
sd          A.13 utt1:  -- [ we, + {F uh, } we ] killed a boar the other day,  /
sd          A.13 utt2: it was, {D you know, } mating with the sows,  /
sd          A.13 utt3: {C and } you can’t use the piglets, {D you know, }  /
%          A.13 utt4: {C so. } -/
```

Here is another example of ’sv’, where Speaker A. is describing his family’s camper, illustrating USE of ’sd’ for a statement evaluating something the listener ’doesn’t get to be expert on’:

```
sd          A.31 utt3:  It’s really nice,  /
sd          A.31 utt4: in fact, it even [ had, + had  ] a little refrigerator,
                 {F uh, } and the whole business.  /
sd          A.31 utt5:  It was quite nice in that respect.  /
sd          A.31 utt6:  {F Uh, } {C and } everything was very convenient /
```

Examples of sv: (topic of the opinion is general: siamese cats)

```
qw          A.11 utt1:   {F Oh. }  {F Uh, } how’s the disposition
                          of your Siamese cat? /

sv          B.12 utt1:  {D Well, } it’s, {F uh, } {D you know } they’re
                        just,  { F uh, } aggressive by nature -- /
...
```

Conversation sw01_4019: talking about rabbits, which neither speaker has as a pet:

```
sv          B.70 utt3: {C and } I would imagine that they don’t  have many more
                       than one to start with, either. /

b          A.71 utt1:  Yeah.  /
sv          A.71 utt2:  {D Well, } rabbits are darling.  /

sv          A.71 utt3:  That would be fun if you could get them trained.  /
sv          A.71 utt4:  Otherwise they’re pretty smelly .
```

Here is an example of ’sv’, where speaker A is talking about his/her opinion on war, something anyone may be ’expert’ on:

```
sv       A.25 utt8: {C and } I believe that the real warfare is not with
             Saddam Hussein, or the North Vietnamese,  /

sv       A.25 utt9: {C but } it’s in spiritual kingdoms, and that the real
             warfare is done, {D you know, } in your prayer closet, on your knees.
```

Some clues for ’sv’ are phrases like the following:

```
I think
I believe
It seems
It’s my opinion that
I mean
Suppose
Of course,
```

```
            impersonal ’we’
            impersonal ’they’ as in ’they say it rains a lot there...’
```

Example using impersonal ’we’ in an ’sv’:

```
sv           B.30 utt1:  {C And, } this is what I find particularly difficult
        in that, { F uh, } if we see injustice, and weather it’s in [ a, + ]
        {F uh, } {D you know, } Chicago, [ [ or, + {F uh, }
```

(These are not infallible heuristics, just helpful indicators).

Song titles, book titles, etc, usually appear in ALL CAPITALS in the transcription and will generally be coded as statements when they appear as in the following:

```
qw           A.107 utt2: [ what kind of music [ is, + does ] + # what # --
%           B.108 utt1:  # [ It, + it, ] # -/
+           A.109 utt1:  -- songs does ] he play?  /
sd           B.110 utt1:  [ Th-, + THIS ] LOVE CUTS LIKE A KNIFE.
```

5.3. Influencing-addressee-future-action

DAMSL	SWBD
Influencing-addressee-fut-actn	
Open-option	oo
Directive	
Info-request	qy, qw, qo, qr, qrr, ^d, ^g
Yes-No-question	qy
Wh-Question	qw
Open-Question	qo
Or-Question	qr
Or-Clause	qrr
Declarative-Question	^d
Tag-Question	^g
Action-directive	ad

DAMSL Open-option directly maps to SWBD-DAMSL "oo". **oo** codes cases which are like commands (’Action-directive’s = **ad**) except that with **oo** the talker offers the hearer multiple options; it comes across as a suggestion.

```
oo           A.3 utt1:  You can go first,  /
oo           A.3 utt2: {C or } I will. /
...
```

```
oo^t           A.1 utt1:  {C And } I guess, the suggestion is that we maybe talk abou
...

oo^t           A.1 utt1:  We could talk about my favorite subject . /
...
```

5.2.1 Action-Directive "ad" (commands, proposals, etc)

DAMSL Action-Directive is coded exactly by SWBD-DAMSL **ad**. It marks imperatives and commands. Because of the nature of Switchboard, most of the imperatives are commands to speak ("Go ahead", "Tell me more about that", etc).

The syntactic realization of **ad** may include imperatives, questions ("Do you want to go ahead and start?"), and standard declarative clauses ("You ought to rent the, {F uh, } F X part one.").

Some examples:

```
ad           A.1 utt1:  Go ahead . [after an overlap] /

aa           B.2 utt1:  {F Oh, } okay .  /
```

_____

```
sd^t           B.2 utt2:  [ I, + I ] think we're started now. /
b          A.3 utt1:  {F Oh, } okay. /
ad       B.4 utt1:  {F Uh, } do you want to go ahead and start? /
```

_____

```
ad           A.95 utt2: you ought to rent the,  {F uh, } F X part one.  /
```

_____

```
ad           A.1 utt1:  Tell me what you like to do. /
```

5.2.2. Info-request (info-questions) (qy,qw,qo,qr,qrr,^d,^g,qh)

The SWBD-DAMSL (qy,qw,qo,qrr,^d,^g) tags are a proper subset of the the DAMSL Info-request tags. qy,qw,qo,qrr are to be used for utterances that are jointly pragmatically, semantically, and syntactically questions. This is another case of "shortcut" tags that encode multiple dimensions; for example qy is used of a question that

1) From a discourse perspective expects a Yes or No (or constrained Other) answer

2) From a syntactic perspective has the attributes of a yes-no-question (i.e. subject-aux inversion, do-support, question intonation etc)

So "qy" would *not* be use of an action directive (command/proposal) that merely takes the *syntactic* form of a question; the following is *not* a "qy", but an "ad":

```
ad  A: Can you pass the salt?
```

What about an utterance that is pragmatically a question but has declarative syntax? These get the ^d "declarative question" label.

*****Coder's Heuristics****

Here's a summary of what markings you should use for different things that may or may not be questions at at least one level.

```
                                           Is it a question at this level?
Type                                  Tag      Prag     Syn
Question                              q          yes     yes
Declarative Question                  q^d        yes     no
Reformulation/Summarization           bf         yes     no
Action Directive (Command/Proposal)   ad          no     yes
Continuer in the form                 bh          no     yes
    of a Rhetorical Question
    (e.g. "oh, really?")
rhetorical question                   qh          no     yes
```

Why does SWBD-DAMSL distinguish wh-questions, yes-no questions, open-ended, and or-questions (qw,qy,qo,qr) where DAMSL doesn't? It is not just because these questions are syntactically distinct. They also have quite different forward functions; a yes-no question is likelier to get a "yes" answer than is a wh-question.

5.2.2.1. qy

---

qy is used for yes-no questions **only** if they both have the pragmatic force of a yes-no-question *and* if they have the syntactic and prosodic markings of a yes-no question (i.e. subject-inversion, question intonation).

```
qy          B.82 utt1: Do you have to have any special training? /
qy          A.1 utt1: Do you know anyone that, {F uh, }[ is, + is ] in a
qy          A.1 utt1:  Okay, {F um, }  Chuck, do you have any pets # there at your h
qy          B.28 utt1:  Does he bite her enough to draw blood? /
qy        B.48 utt1:  Is that the only pet that you have? /
qy          A.55 utt2: {D So } have you tried any other pets? /
qy          A.96 utt3: Do you? /
```

Yes-no questions that are pragmatically questions but have declarative syntax are marked with **^d**. Yes-no questions that are syntactically (in form) questions but do not rhetorically function as questions ("rhetorical questions") are marked either as **qh** or **bh**, depending on whether the rhetorical question is functioning as a backchannel. See the other sections for examples of each of these other kinds of "questions".

5.2.2.2 qw

Coder’s Heuristics

Wh-interrogative questions. These must have subject-inversion. "Echo-questions" with wh-in-place are considered "declarative questions" (marked with ^d, see below).

```
qw          B.94 utt1:   {F Um, } what cities are they looking at? /
qw          B.3 utt2:  How old are your children? /
qw          B.48 utt1:  {D Well } what other long range goals do you have...
qw          A.1 utt1:   {D So, } who’s your favorite team? /
qw          A.1 utt2: What kind of pets do you have? /
```

5.2.2.2b qw^d

These are usually but not always wh "echo-questions" (‘You said what!?’)

```
qw^d          B.22 utt1:  [ {C And, } + {C and } ] you say you’ve had him how long?
```

____

```
qw^d          A.3 utt2: {D So, } when you say the morning news, or evening news or n
```

5.2.2.3. qo (open-ended questions)

Coder’s Heuristics

These are mostly of the "how about you" variety; "qo" is meant to address the kind of questions which we think place few if any syntactic constraints on the form of the answer.

```
qo          B.4 utt1:  How about you? /
qo          B.31 utt3:  # What do you think? # /
qo          B.18 utt1:  How about yours? /

qo          Speaker B:   {D So } what are your opinions on it? /
                         [HYPOTHETICAL EXAMPLE]
qo          A.1 utt1:   What do you think about the benefits in jobs? /
qo          A.7 utt1:   How about your community? /
```

5.2.2.4. qr "or"-questions

Coder's Heuristics

examples:

```
qr          B.50 utt1:  {D Well, } do you live, [ [ you, + you ] + ] in a house,
                        or a  place where you, {F uh, } -/
qr          B.95 utt1:  # {D Well } # do you all work for T I, or for, -/
qr          B.36 utt1:  # {D Now, } # [ are they, + are they ] rehabilitative
                            [ or, + or ] not. /
```

One problem with or-questions is that the listener often interrupts before the or clause is complete and answers the or-question as if it were a yes-no question about the first clause. For example

```
qr      B60 utt1:  Did you bring him to a doggy obedience school or --

nn      A61 utt1:  No --  /

+       B62 utt1:  -- just --

sd^e    A63 utt1:  -- we never did. /

+       B64 utt1:  -- train him on your own   /
```

We counting this as a qr since the speaker goes on to finish his qr, even though the listener answers it immediately as a yes-no question. Our current viewpoint is that if there's a conflict between labeling "what the speaker thinks" and "what the hearer thinks" go with whichever coding is more informative for the reader, which in this case is the speaker-labelling (because if you were reading the transcript you could figure out that a qr followed by a "No" answer means that the listener misinterpreted. But if you labeled it the other way (i.e. as a "qy") then it would be harder to figure out that the speaker was thinking of the utterance as an or-question.

5.2.2.5 qrr "or-clause tacked on after a y/n question"

Coder's Heuristics

These are used when you think the speaker tacked on an or-clause to what had been a yes-no question, so "qrr" marks a sort of "dangling or-clause", e.g. B.18.utt2.

```
qy          B.18 utt1:  # [ Do you watch, + # do you watch ] [ the network, +
                         {D like } major network ] news,  /
qrr          B.18 utt2: {C or } do you watch {D like } --

sd          A.19 utt1:  [ Just the # regular channel # -- +
```

```
+          B.20 utt1:  -- # the MACNEIL LEHRER HOUR? # /

sd          A.21 utt1:  -- just channel eight. ] /
```

When the speaker uses the word "or" after a **qy**in a slash-unit by itself at the end of a turn, it is coded as a **turn-exit** (i.e. **%**):

```
qy*      B.64 utt1:  {F Uh, } is that the crime  /  [[*listen]]
qy       B.64 utt2:  {C and } it's already,  ((   ))  some chart and
              determine the punishment,  /
%      B.64 utt3: {C or. } -/
```

5.2.2.6 ^d "declarative questions"

These labels are in an independent dimension from the other question labels (qy,qw,qo,qr,qrr). Like some of the other SWBD-DAMSL "extra dimensions", these are primarily designed to code form.

Declarative questions (^d) are utterances which function pragmatically as questions but which do not have "question form". We don't know if declarative questions will have different conversational function than non-declarative question (although see Weber 1993 for thoughts on this), but we definitely expect them to be useful for ASR language-model purposes.

Declarative questions normally have no wh-word as the argument of the verb (except in "echo-question" format), and have "declarative" word order in which the subject precedes the verb. See Webber 1993 Chapter 4 for a survey of declarative question and their various realizations.

Declarative questions *may* have rising "question-intonation". The "declarative" tag is added solely based on form. This does not mean that the intonation of the question is irrelevant. We are marking the prosodic features of each utterance in Switchboard in another, distinct database.

Coder's Heuristics

These are all ^d (declarative questions): (B.46.utt1 is an example of a declarative question with a wh-word)

```
qy^d          B.44 utt1:    {D So } you're taking a government course? /
qw^d      B.46 utt1:  At what?  /
qy^d         B.46 utt2:  The university? /
qw^d          B.22 utt1:  [ {C And, } + {C and } ] you say you've had him how long?
qy^d          A.1 utt3:  I don't know if you are familiar with that./
qy^d          A.3 utt1:  {C But } not for petty theft?
qy^d          A.65 utt1:  {D Well, } I guess we'll get pretty good news coverage
                     in a couple of years when you host the, { F uh, }
                     summer olympics . /
```

Or the following:

```
qy^d          B.2 utt2: You're asking what my opinion about,
```

```
 ny           A.3 utt1:  # Yeah. # /

  +          @B.4 utt1:  # whether it's # possible  to have honesty in government.
```

Or here's another one:

```
qy^d          A.64 utt2: you must be a T I employee. /
```

However, if the statement has an "ellipsed" aux-inversion at the beginning, we don't code it as a declarative question (following Weber 1993).

```
qy          B.44 utt1: Worried that they're not going to get enough
        attention? /
```

5.2.2.7 question tags (^g)

A 'tag' question consists of a statement and a 'tag' which seeks confirmation of the statement. Because the tag gives the statement the force of a question, the tag question is coded 'qy^g'. The tag may also be transcribed as a separate slash unit, in which case it is coded '^g'.

Coder's Heuristic

A question designed to check whether the listener understands what the speaker's point is should be distinguished from a question tag. Listener may respond affirmatively that s/he understands what was said without implying agreement. "understand what I'm saying" and thus respond affirmatively to an 'understanding check' but disagree with speaker's statement. The appropriate response to a tag question, on the other hand, confirms the *statement*.

The appropriate code for an understanding check is "qy"

The appropriate code for the response, like the response to a tag question, is usually ny or nn. The appropriate response to an understanding check is also 'ny' or 'nn.

In answering a true tag, you are confirming or disconfirming the statement that precedes it.

In answering a question about 'understanding-check', listener is not taking any position on the statement that preceded it. S/He is merely indicating that the statement was understood.

Tag questions all have either an aux-inversion at the end (don't you? doesn't it? isn't he? aren't you?) which (almost always) reverses the polarity of the auxiliary in the matrix statement, or a one-word tag like ", right?" or ", huh?".

Here are some examples of ^g (tag questions): single-word tag:

```
qy^g      A.39 utt2: {F Uh, } I guess a year ago you're probably watching C N N a lo
```

unreversed polarity, with subject-aux inverted tag:

```
qy^g@     @B:  {D So } you live in Utah do you? /
```

reversed polarity, with subject-aux inverted tag:

```
qy^g       A.27 utt1:  That's a problem, isn't it? /
qy^g       B.54 utt1:  # {C But } that doesn't eliminate it, does it? # /
```

tag in single slash unit:

```
sd      A.1 utt 1:      Well, Hank Williams is one we forgot about.  /
^g      A.2 utt 2:      Right?  /
```

____

```
sd      A.13 utt2: as a matter of fact, I want to think they took the top
             managers first,  /
^g      A.13 utt3: isn't that a fact?  /
```

5.2.2.8 Rhetorical questions qh

Coder's Heuristics

Rhetorical questions are 'qh' (question-rHetorical) as in the example(s) below :

```
ad          A.63 utt2: {C and } think [ what, + what's ] it going to be
        like for [ [ my, + my youngest, ] + [ an + ] my oldest ] son, when he goes t
qh          A.63 utt3: What's going to happen?  /
sd          A.63 utt4: {E I mean } [ I, + I'm ] afraid for him to go. /



+          B.52 utt1:  -- like, {D you know, } the old day with the rack.  /
        sv(^q)     B.52 utt2:  [ We, +  they're ] going to say, Okay, you're guilty
        you have to pay Kuwait four million dollars.  /

qh          B.52 utt3: {D Well, } whose going to really make them. /

b          A.53 utt1:  Yeah. /

sv          B.54 utt1:  Nobody. /

b          A.55 utt1:  Yeah, /
```

Coder's Heuristic

Be careful not to confuse rhetorical questions with 'bh', backchannels which take the syntactic form of rhetorical questions. Unlike rhetorical questions, backchannels lack semantic content:

```
bh      B.18 utt1: {F Oh, } really? /
```

5.2.3 Committing-speaker-future-action

```
       Committing-speaker-future-action
            Offer                          co
            Commit                         cc
```

The SWBD-DAMSL labels "co" maps directly to DAMSL "Offer" and "cc" maps directly to DAMSL "Commit", except for one important caveat.

The caveat is that the SWBD-DAMSL tags assume that Offers and Commits only occur in the context of some sort of negotiation (in a weak sense); that not every future action ("I'm going to try out for crew next season") is an Offer.

That is, where Allen and Core say that

> "the defining property of utterances with this aspect is that they potentially commit the speaker (in varying degrees of strength) to some future course of action." (p 11)

we assume this means "not all future courses of action" (since speakers often discuss "what they plan to do this weekend") but only those involving the conversational partner in some way. Here's an example of **cc** where a speaker commits pushing a button:

```
^h           A.5 utt1:  Let me see,  /
sd^t           A.5 utt2: I don't know if that took or not,  /
cc^t           A.5 utt3: I'll do it again. /

b           B.6 utt1:  Okay. /
```

The distinction between Offer and Commit depends on "whether the utterance's commitment is conditional on the listener's agreement or not." (p 11). So here's an example of an Offer (**co**):

```
co           A.47 utt2: we could talk about some of the long range goals  /
```

Here's a other one with an Accept (**aa**):

```
co           A.61 utt1:  I have a recipe if you want . /

aa           B.62 utt1:  Okay,  /
aa           B.62 utt2:  sure, [ su-, + ] /
```

When the speaker is suggesting that the speaker is about to do something in a polit way that gives the listener a chance to say "no" in a sort of default way, this is "co":

```
co    Let me ask, by the way, just for the record.  /
co    Let me turn off my stereo here
co    Let me push the button. /
co    Let me change my channel,
co    Let me see if that clears this up.  /
```

```
co    let me try it again because usually , {F um. } -/
co    Hang on let me check  (( on it )) .  /
```

5.2.4 Other-forward-function

```
       Other-forward-function
            Conventional-opening            fp
            Conventional-closing            fc
            Explicit-performative           fx
            Exclamation                     fe
            Other-forward-function          fo,ft,fw,fa

        fp  oPenings (hi)
        fc  Closing  (bye)
        ft  thanks
        fw  you're welcome
        fa  apologies (not the "I'm sorry" of sympathy, just the apology) "excuse me


        fp  "hello"
        fe  "ouch"
        fe  "oh, golly"
        fx  "you're fired"
```

5.2.4.1 Openings 'fp'

Openings (**fp**)have often been cut out of switchboard, but some of them still remain; they may continue on for more than one slash unit. See Schegloff (1968).

```
fp          A.1 utt1:   Hi, Wanet <>.  /
fp          A.1 utt2: How are you? /
_______
fp          B.2 utt1:  I'm doing fine.  /
```

5.2.4.2 Closing 'fc'

Closings (**fc**) are much more common. They also often continue on for well more than one slash unit:

```
fc          B.150 utt2:   {D Anyway, } it's been nice talking to you. /
fc          A.151 utt1:  Yeah,  /
%          A.151 utt2: {D well. } -/
%          B.152 utt1:  {C And, } {F uh, } -/
```

```
fc          A.153 utt1:  {D Well } good luck with [ the, +  the ] new kid. /
ft          B.154 utt1:   Thank you,    /
fc          B.154 utt2:  [ [ she's, +  it, ] +  she's ] good.  /
```

Our current policy is to mark every slash-unit in the entire closing sequence as (solely) **fc**. That is, once the 'fc' sequence begins, in general, we will code the sequence as 'fc' until the actual closing of the conversation. These need to be looked at further to re-examine the internal structure of these closings (in particular with regard to Schegloff and Sacks 1973).

5.2.4.2 Thanks 'ft'

Mostly "thank you". Don't forget we don't mark these if they occur in the closing; then they get marked as **fc**.

5.2.4.3 Welcome 'fw'

Nobody says "you're welcome" any more. What they say is:

```
../sw02utt/sw_0212_2275.utt:fw          A.153 utt1:  Uh-huh. /
../sw06utt/sw_0634_2027.utt:fw          B.108 utt1:  # Okay,  /
../sw07utt/sw_0709_2952.utt:fw          A.211 utt1:  Uh-huh.  /
../sw08utt/sw_0871_2930.utt:fw          B.128 utt1:  You bet,  /
../sw10utt/sw_1033_2723.utt:fw          A.147 utt1:  Yeah. /
```

5.2.4.3 Exclamation 'fe'

These are mostly generated by the following grammar:

```
(oh|well|i mean|NIL) (gosh|goodness|boy|good grief|jeez|heavens|shoot|gee whiz)
```

5.2.4.3 eXplicit performative 'fx'

Not very many. All "i bet you", "i wish you", or "i recommend". Here's all of them:

```
{D Well } I wish you very good luck with it
I bet you can't guess .
I am going to bet you that is a lily. Because it is,
{F Oh, } [ I bet you those are, +  I bet you what those things are, ] {F uh, } is a
I bet you it is a Dutch iris.
I am going to bet you that,
```

```
I will bet you those are Dutch iris.
I do recommend the  (( for savings ))  bit.
```

5.2.4.3 apologies ’fa’

Coder’s Heuristics

"Excuse me" was coded as ’fa’ if it followed something for which the speaker was apologizing, such as a cough or an interruption. ’Excuse me’ was coded as ’co’ if it preceded something the speaker was negotiating permission to do in advance of doing it. If the speaker is asking permission to do something (like below, "excuse me just a second") it is ’co’ If the speaker is apologizing for something s/he just did, (sneezing, coughing, etc), it is ’fa’

```
b           B.30 utt1:  Yeah,  /
ba          B.30 utt2: that is nice. / @@A:  Yeah   /
qy^d     B.30 utt3: {E excuse me, }  it sounds like we both have colds. /
ny           B.31 utt1:  Yeah,  /
_______
sd          A.63 utt1:  {D All right, } {F uh, } {D you know, }  [
             there’s bumble bee patterns + --
b           B.64 utt1:  Uh-huh. /
+           A.65 utt1:  -- [ there’s , +   {E excuse me. }  {F
             Uh, } there’s ] bumble patterns, ] there’s mosquito patterns, there’s
             wasp patterns, there’s grub patterns
```

---

6. Backwards-Communicative-Function

DAMSL	SWBD
Backwards-Communicative-Function	
Agreement	
Accept	aa
Accept-part	aap
Maybe	am
Reject-part	arp
Reject	ar
Hold before answer/agreement	^h
Understanding	
Signal-non-understanding	br, br^m
Signal-understanding	
Acknowledge	b,bh
Acknowledge-answer	bk
Repeat-phrase	^m
Completion	^2
Summarize/reformulate	bf
Appreciation	ba
Sympathy	by
Downplayer	bd
Correct-misspeaking	bc
Answer	DEFAULT-for-qw,ny,nn,na,nd,ng,no,sd^e,sv^e,^h
Yes answers	ny
No answers	nn
Affirmative non-yes answers	na
Negative non-no answers	ng
Other answers	no
Expansions of y/n answers	^e
Dispreferred answers	nd

The backwards-communicative function breaks roughly down into Agreements, Understandings, and Answers.

6.1 Agreements aa,aap,am,arp,ar,ah

DAMSL	SWBD
Accept	aa
Accept-part	aap
Maybe	am
Reject-part	arp
Reject	ar
Hold before answer/agreement	^h

The Agreements (Accept, Reject, Partial Accept etc) all mark the degree to which speaker accepts some previous proposal, plan, opinion, or statement. This is a generalization over the use in Allen and Core (1997), which seems to reserve Agreements for accepts or rejects of proposals, not statements.

An example of **aa** in accepting a proposal ('ad'):

```
ad          A.1 utt1:  Go ahead . [after an overlap] /

aa          B.2 utt1:  {F Oh, } okay .  /
```

Some examples of **aa** marking agreements with previous opinions:

```
aa          A.19 utt1:  # that's # what I was thinking too. /

__________
aa          A.41 utt2:  Yeah  /
aa          A.41 utt3: that would be a real good idea.  /

__________
aa          B.146 utt1:  Yes,  /
aa          B.146 utt2: {F uh, } [ that sounds like a good, +
                that sounds like the right ] theory. /

__________
sv          B.40 utt3: That was a really good movie. /

aa          A.41 utt1:  It sure was.  /
sv          A.41 utt2: {C And, } {D you know, }  the second time you see it, you
                        understand more subtleties in it.  /
sv          A.41 utt3:   There are a number of good movies like that. /

__________
sd          B.70 utt5: I could just sit there all day and look at the
             scenery . /

aa          A.71 utt1:   Yes. /
aa          A.73 utt1:   [ I, + I ] agree.  /
sd          A.73 utt2: [ I can, + I can ] do that  too,  /
```

Coder's Heuristics

We have aa's that are one-liners:

```
Exactly!
Definitely.
Yes. (not 'yeah')
That's a fact.
That's true.
True.
```

"Yeah" as 'aa':

Some 'yeah' s (and to a lesser extent, some uh-huh's) are 'aa' and some are not. They are not 'aa' if they occur alone, without some second utterance to support the idea of agreement.

We will not code a "yeah" or "uh-huh" as 'aa' unless it is followed by an additional utterance indicating agreement:

```
sd           B.38 utt2: I also like jazz. /

aa          A.39 utt1: Yeah.  /
sd           A.39 utt2: Me [ too, + too. ]  /
```

If there is a second statement, and it is brief, you may code the two utterances as "aa'

```
aa      Speaker1 utt1: Yeah.
aa      Speaker1 utt2:  You're right.
(HYPOTHETICAL Example)
```

If there is a second statement and it is more complex, code the second statement as sd or sv, as the case may be.

```
sv      Speaker1 utt1:  Clinton's an idiot.
aa      Speaker2 utt1:  Yeah.
sd      Speaker2 utt2:  He's an idiot because of his dumb welfare policy.
(HYPOTHETICAL Example)
```

Here is an example of a "yeah" followed by a second statement which is NOT indicating 'agreement' in the sense required to code 'aa' because it is not showing agreement but rather just continuing on with new information on the same topic:

```
sv      A.1 utt3: I think it's, {F uh, } refreshing to see [ the, + {F uh, }
         the  ] support that the President got from the American people. /

b       B.2 utt1:  Yeah,  /
sd      B.2 utt2: [ [ [ it, +  we, ] +   I, ] +  I ] read an interesting
```

* More Coder's Heuristic's*

Thinking alike generally constitutes agreement; being alike may not. This is demonstrated in the following HYPOTHETICAL examples:

```
sd      Speaker1 utt 1: I have a Mercedes.
sd      Speaker2 utt 1: Me, too.
```

```
__________
sd      Speaker1 utt1: I like Mercedes.
aa      Speaker2 utt1:  Me, too.

__________
sd      Speaker1 utt1: I think Mercedes are great cars.
aa      Speaker2 utt1:  Me, too.
```

Here’s a reject of a previous opinion:

```
               , + I ] don’t particular like the fact that it’s the military,
                  {D you know, }  /
sv          B.37 utt4: (( )) {C and } the whole point of the military is to kill
 people essentially. [ As, + as ] an instrument of U S # policy. # /

ar          A.38 utt1:  # {F Oh, } no,  /
ar^r          A.38 utt2: # no,  /
ar^r          A.38 utt3: no.  /
sv          A.38 utt4:  It’s to defend the nation against external evils. /
```

A negative response to a question, statement or proposal is not necessarily a ’reject’. If the previous statement is phrased in the negative, a ’no’ could be an agreement, as in the following example:

```
sd      B.48 utt1: {E I mean } the stuff I’ve read recently in
             Technology Re view basically indicates that acid rain may be a
             little bit, {F uh, } overstated.  That a lot of the die off
             they’ve seen in forests may not really be due to acid rain at all.  /
%       B.48 utt2: {F Um, } ye-, - /
sd      B.48 utt3: I’m not an expert. /

aa      A.49 utt1: Yeah,  /
aa      A.49 utt2: no.  /
```

And a speaker can change his/her mind by accepting, then rejecting:

```
sd          B.26 utt1:  I don’t think women look good with muscles. /

aap          A.27 utt1:  Up to a point. /

sd^r          B.28 utt1:  Up to a point,  /
ar          B.28 utt2: no,  /
aa          B.28 utt3: [ m-, + ] yeah, . /
```

Final Coder’s Heuristics

Don’t use aa to code the ’yeah’ "incipient speaker-shift" that we have been trying to code. Use b for that for now.

```
+*          A.21 utt1:  {F uh, } {D you know, } I don’t really ] feel as though
                          I’ve a gotten sufficient, {F uh, } {D you know, } dose
                          of news that way. /

b           B.22 utt1:  Yeah.  /
sd          B.22 utt2:  A lot of my information comes from several sources.  /
sd*          B.22 utt3: Probably pretty high up on the list is National
                                  Public Radio.
```

Very few of the sentences with "maybe" in them are actually MAYBE's. There were no MAYBE's in the first 25 conversations. Here are two examples:

```
sv          A.39 utt1:  #  A shotgun hurts worse # than a pistol does. /


am          B.40 utt1:  {F Uh, } yeah.  /
"          B.40 utt2:  I suppose.  /
sd          B.40 utt3: I never got shot with either one.  /
```

____

```
sd          A.105 utt1:  My husband feels that they'll come and collect
        everybody's guns. /

b          B.106 utt1: Yeah.  /
am          B.106 utt2: I guess that could happen.  /
```

____

```
sd          B.145 utt2: {C so } I can't complain too much. /

b          A.146 utt1:   Yeah,  /
am          A.146 utt2: I guess so.  /
am          A.146 utt3: I don't know.  /
```

____

```
sv          B.2 utt3: {F Uh, } {C but } I suspect [ it, + it ] very much
              depends upon the job. /

b          A.3 utt1:  Huh-uh. /

am          B.4 utt1:  Maybe.  /
sv          B.4 utt2:  There are some jobs where I guess it doesn't really, /
```

6.2 Understanding br,b,bg,b^m,^2,bf,ba,by,bd

DAMSL	SWBD
Understanding	
Signal-non-understanding	br, br^m
Signal-understanding	
Acknowledge	b,bh
Acknowledge-answer	bk
Repeat-phrase	^m
Completion	^2
Summarize/reformulate	bf
Appreciation	ba
Sympathy	by
Downplayer	bd
Correct-misspeaking	bc

This class includes what markers of understanding at various levels, including what Yngve (1970) called "backchannels", ("continuers" or "assessments" in the CA literature), as well as markers of misunderstanding like requests for repeat and corrections of misspeaking ("next-turn-repair-initiators"), and others. See Schegloff (1982) and Jefferson (1984) for surveys of some of these.

6.2.1 Signal-non-understanding br and br^m "requests for repeat"

**Mapping to DAMSL heuristic*:* All br's are also ACTION-DIRECTIVE.

```
br B74 utt1:  Invisible what?   /
```

Another example:

```
qy^d          A.64 utt2: you must be a T I employee. /

br^m          B.65 utt1:  You must be what? /
```

6.2.2 Signal-understanding b,bh,^m,^2,bf,ba,by,bd,bc

SWBD-DAMSL has more sub-types of these than SWBD because they account for 25% of our utterances.

6.2.2.1 Acknowledge "b"

Your basic 'b' is what is usually referred to in the CA literature as a "continuer". Of the approximately 300 types (35,827 tokens) of pure **b** the most common ones are the following:

```
38% uh-huh
34% yeah
9%  right
3%   oh
2%   yes
2%   okay
2%   oh yeah
1%   huh
1%   sure
1%   um
1%   huh-uh
1% uh

[Less than half a percent each:]
     really
     no
     oh uh-huh
     oh okay
     oh really
     yep
     i see
     well yeah
     all right
     oh i see
     -- yeah
     oh yes
     uh yeah
     yeah --
     um yeah
     you know
     so yeah
     um uh-huh
     ooh
     oh no
     hm
     oh sure
     that's right
```

Coders Heuristics

Our various experiments on marking "incipient speakership", i.e. the "yeah" that people use to mark the fact that they are about to speak, have not worked well. So for now, mark those "yeah"s with the default backchannel marker ("b").

```
+*          A.21 utt1:  {F uh, } {D you know, } I don't really ] feel as though
                          I've a gotten sufficient, {F uh, } {D you know, } dose
```

```
                         of news that way. /  *[[needs --]]

b           B.22 utt1:  Yeah.  /
sd          B.22 utt2:  A lot of my information comes from several sources.  /
sd*          B.22 utt3: Probably pretty high up on the list is National
                                  Public Radio.    *[[needs --]]
```

6.2.2.1b Acknowledge "bh"

"bh" is a continuer which takes the form of a question. (We are marking these distinctly because we suspect that they will mess up the prosodic utterance detector if they are just thrown in with the "b"s, since they have question intonation.)

The most common is "Oh, really?"; here's some counts (out of ~740 bh's from the first 755 conversations)

```
  141 {F Oh, } really?
  103 Really?
  39 Is that right?
  21 {F Oh, } yeah?
  15 {F Oh, } is that right?
  14 Do you?
  12 Is it?
  11 {F Oh } really?
  10 {F Oh, } did you?
  10 Are you?
   8 Yeah?
   6 {F Oh, } have you?
   6 {F Oh, } do you?
   6 No?
   6 Did you?
   5 {F Oh, } are you?
   5 Was it?
   5 Have you?
   4 {F Oh, } is it?
   3 {F Oh, } you do?
   3 Isn't that interesting?
   3 Isn't that amazing?
   2 {F Oh, } it does?
   2 {F Oh, } do they?
   2 {F Oh, } are you really?
   2 isn't that funny?
   2 You think?
   2 You think so?
```

Coders Heuristics

35% of the time (in the first 755 conversations), these backchannel questions get answered with "yeah". Mark the answer **ny**.

```
sv          A.25 utt1: It was funny.  /
```

```
sd          A.25 utt2: [ There were, + they ha-, ] {F uh, } a fireworks display at h

bh          B.26 utt1:    {F Oh, } yeah? /

ny^m          A.27 utt1:   Yeah,  /
sd          A.27 utt2: {C and } some paper or something in the Super Dome up in the
laughter> on fire.
/


.......

sd          A.19 utt1:  {C And } this lady, you would think it was her own. /

bh          B.20 utt1:  Really? /

ny          A.21 utt1:  Yeah.  /
sd          A.21 utt2:  She's real good. /
```

6.2.2.2 Acknowledge-Answers bk

These are acknowledgements of answers to questions. Thus, they follow a question + answer sequence. 'bk' is almost always "Oh, okay" or "Oh, I see." (This is the "New information 'Oh'", see Schiffrin 1987). Sometimes 'bk' may be simply "okay." Out of the 1339 **bk**'s in the complete 1155 conversations:

```
 418 okay
 284 {F oh, } okay
 144 oh
  48 {F oh, } I see
  48 I see
  35 uh-huh
  18 Yeah
  14 okay.
  11 {F oh, } yeah
  11 right
  11 All right
   9 {F oh, } uh-huh
   9 {F oh, } okay.

qw          A.29 utt2:  {C But, } {F uh, } {F uh, } I was just curious,
        what, {F uh, } part of the country. -/

sd         B.30 utt1:  {F Oh, } Stockton. /

bk         A.31 utt1:  {F Oh, } okay. /
```

```
nn          A.123 utt1: No,  /
sd^e        A.123 utt2: I don't watch T V much at all. /

bk          B.124 utt1:  Okay. /
```

```
qy          B.74 utt2: Were they religious? /

ny         A.75 utt1:  Yes. /

bk         B.76 utt1: {F uh, } I see.  /
```

The bk 'acknowledgement' of answer may not be contiguous with the initial utterance encoding the answer to the speaker's question. Example:

```
qw^t           B.174 utt2:  How'd you get involved in this research? /

sd           A.175 utt1:  {F Um, } I worked at T I for a while,  /
sd           A.175 utt2:  {C but } then my brother-in-law works there,  /
sd           A.175 utt3: {C and } he got me into it. /

bk           B.176 utt1:  {F Oh, } I see.  /
```

But a preceding question+answer pair is *required* before the label 'bk' applies:

```
qw           A.83 utt1:  {E I mean } {C but } [ where are they, +  where are they
, ]  /
qw           A.83 utt2: [ what, + what ] is their location,  /
qy           A.83 utt3: is it, {F uh, } Asian  /
qrr           A.83 utt4: or is it European  /
qw           A.83 utt5: {C or } who, -/

nn           B.84 utt1:  No.  /
nn^r           B.84 utt2:  No,  /
nn^r           B.84 utt3: no.  /
sd           B.84 utt4:  Nissan is Japanese. /

bk           A.85 utt1:  {F Oh, } it is Japanese. /
```

6.2.2.3 Repeat-phrase b^m

In SWBD-DAMSL the "mimic-other-speaker" tag is in the "Form" dimension and so it's orthogonal to all other tags. This is because, since our main focus is speech-recognition, recycling of lexical material is something that we place emphasis on marking.

So the way SWBD-DAMSL codes the Backwards function "Repeat-phrase" is combining b and ^m: b^m. There are 695 of these in the 1155 conversations.

```
qw           B.20 utt1:  {D Well, } [ how, + how ] old are you? /

sd           A.21 utt1:  I'm twenty-eight. /

b^m           B.22 utt1:  Twenty-eight.  /
bk           B.22 utt2:  Okay,  /
sd           B.22 utt3: I'm twenty-three. /
```

6.2.2.4 SUMMARIZE-REFORMULATE bf

This is a new subtype of Signal-Understanding in SWBD-DAMSL which isn't in March 21, 1997 DAMSL. A bf reformulation is used when one speaker is proposing a summarization or paraphrase of another speaker's talk, as in A.58:

```
sv          B.53 utt1:  {C And } you need a special nursing home for that./
sv          B.53 utt2:  You need one that has a unit that's locked where
              they are not able to get out and roam around -- /

b          A.54 utt1:  Yeah. /

sv          B.55 utt1:  -- {C and } you need people who are trained for
           that # type # --

b          A.56 utt1:  # Right. # /

+          B.57 utt1:  -- of problem. /

bf          A.58 utt1:  Who know what they're doing with that. /

aa         B.59 utt1: Yeah  /
```

bf is used when it summarizes the *other* speaker's point: A.9 utt1 below is *not* a bf but an sv, since A is summarizing her/his *own* argument.

```
sd          A.5 utt2: we're not being tested for drugs at all, {F uh, } /
sd          A.5 utt3: our policies and procedures manual, {F uh, } the
                furthest it goes about drugs is in [ the, + kind of the]
                miscellaneous section, or --

b          B.6 utt1:  Uh-huh. /

+          A.7 utt1:  -- it's reasons for immediate dismissal,  /
sd^q          A.7 utt2:  it says, use of narcotics on company premises. /

b          B.8 utt1:  {F Um. } /

sv           A.9 utt1: {C So } that's pretty general,/
```

Coders Heuristics

We don't mark summarizations/reformulations of one's own argument since they don't have as obvious a discourse function as summarization of other-talk; summarizations of other-talk function pragmatically as questions.

Reformulations are often (about half of the time?) marked by starting with one of the following: (with statistics out of the 660 bfs in the first 755 conversations coded:)

```
             In utterance   Starts utterance
```

```
  you/you're 33%              7%
  {C so}     13%              10%
  {F oh}     8%               7%
```

About 2% of the reformulations have 'you mean' somewhere within the utterance:

```
+           B.30 utt1: makes you cry it sounds so sad # .  /
sv           B.30 utt2: {E I mean } you d-, # -/

bf           A.31 utt1: #  That's the # kind you like you mean? /

aa           B.32 utt1: Yeah.  /
```

Some assorted examples from different conversations:

```
  bf           B.42 utt2: {C so } it's fairly safe. /
  bf           B.76 utt1: {F Oh, } {C so } they don't go to school.  /
  bf           B.6 utt1:  # {F ((Oh)) } they thought it was too much of a
  bf           B.10 utt2:  You're very close actually.  /
```

An example of a syntactic question rather than a 'bf':

```
sd      A.43 utt4: [ I, + I ] [ d-, + don't ] feel comfortable about
        leaving my kids in a big day care center, [ but, + ] simply because
         there's so many kids and so many   , -/

qy           B.44 utt1: Worried that they're not going to get enough
         attention? /

ny           A.45 utt1: Yeah,  /
sv^e         A.45 utt2: {C and, } {F uh, } {D you know, } colds and
                 things like that  get --
```

Coder's Heuristic for Response to 'bf'

Reformulations 'bf'(and Completion ^2, see 6.2.2.5, below) function as understanding-checks; they are pragmatically questions (the implicit question being something like "is this an acceptable summary of your talk?") (though it is not syntactically formed as a question). They often get responses which indicate understanding. When this occurs, we will code the response which agrees with and/or accepts the understanding check as 'aa' and the response which indicates the reformulation was not accurate as 'ar' ('reject'). Partial acceptance 'aap' and partial rejections 'aar' are possible.

So the "yeah" response which often follows a "bf" is an "aa", not a "b" backchannel. or an 'ny' 'aNswer-Yes'.

```
bf           A.31 utt1: #  That's the # kind you like you mean? /

aa           B.32 utt1: Yeah.  /
```

6.2.2.5 Completion ^2

---

^2 marks Completions (also called "collaborative completions"). It can be combined with other labels or used alone:

```
sv          A.23 utt3:  In other words, [ you'd have to, + you'd have to ]
                         murder more than one other person --

^2           B.24 utt1:  Besides him. /
```

Completions '^2' (like 'bf') also function as an understanding-check. They often get responses which indicate understanding. When this occurs, we will code the response which agrees with and/or accepts the completion '^2' as 'aa' and the response which indicates the completion was inaccurate as 'ar' ('reject'). Partial acceptance 'aap' and partial rejections 'aar' are possible.

```
^2          B.92 utt1:  Educational or vocational training or something. /

aa          A.93 utt1:  Yeah.  /
sv          A.93 utt2:  Something that's going to help them along the way. /
```

6.2.2.6 BACKWARDS-ATTITUDE ba,by,bd

These are in SWBD-DAMSL but not in DAMSL. ba is especially common.

6.2.2.6.1. ba. Assessments/Appreciations:

A backchannel/continuer which functions to express slightly more emotional involvement and support than just "uh-huh". Some examples:

```
ba          A.27 utt2:  I can understand that. /
ba          A.31 utt1:   That would be nice. /
ba          B.40 utt1:  I can imagine. /
ba          B.38 utt2:  It must have been tough.  /
ba          B.13 utt3:  That is good.  /
ba     A29 utt1:  {F Oh, } {F oh, } great.   /
ba     A11 utt1:  {F Oh, } he'll be delighted. /
ba     B22 utt1:  #That's great.# /
ba     B30 utt1:  That's great! /
ba     B50 utt1:  {F Oh, } that's great. /
ba     A37 utt1: That's probably a good idea.   /
ba     B32 utt2: that makes sense. /
ba     A.35 utt1:  You bet. /
ba     B.98 utt1:  {C And, } {F uh, } I know exactly what you mean. /
```

And in context:

```
sd         B.13 utt1:  -- {F uh, } especially [ if, + if ] it's after an
           acute illness.  /
```

```
sd          B.13 utt2:  To get over a, - /
sd         B.13 utt3: {C or } to rehab after, {F uh, } an illness. /

aa          A.14 utt1:  That’s true.  /
ba          A.14 utt2:  I never thought of that.  /
```

(Note: James A. suggests "ba"s may also have a forward function as ASSERT, But some of them may not (Does "I can imagine." ?). Confirm these with DAMSL folks.).

(Note: James A. also suggests: could an Assessment also appear as a forward function? ("Here’s a nice idea/ let’s go to the beach") keep our eyes open for this)

6.2.2.6.2 by and bd "sYmpathy" and "Downplayers" of sympathy and compliments

---

These are subtypes of BACKWARDS-ATTITUDE which express not just acknoweldge or understanding, but also further emotional involvement.

```
by   A.44 utt1:  I’m real sorry. /
```

Actual apologies (for doing something), as opposed to markers of sympathies, are tagged as "fa", see above.

bd is any downplayer that speakers use to respond to apologize.

```
bd   B.45 utt1:  That’s all right. /
```

Downplayers are also used to respond to compliments. In the example below, speaker B has just finished going into detail about the topic under discussion, showing his obvious expertise:

```
sv           A.24 utt1:  {D Well, } [ you are, + you are ]  well versed on
                the subject, I tell you. /

bd          B.25 utt1:  {D Well, } I don’t know. /

sd          A.26 utt1:  This is not one of my better ones.  /
```

Most common types: (counts from the 1155 conversations)

```
  19 that’s okay
   7 no
   5 that’s okay
   5 that’s all right
   4 okay
   3 {F oh, } that’s okay
   2 it’s okay
   2 Uh-huh
   2 No
   1 {F um, } {C but } it’s okay
   1 {F oh, } {D well. }
```

```
1 {F oh, } {D well, } I guess I'll get over it
1 {F oh, } {D well } that's okay, {F um, }
1 {F oh, } you're not
1 {F oh, } that's okay.
```

6.2.2.6.3 bc "Correct-misspeaking-by-other-speaker"

These aren't very common in this genre but they can be amusing:

```
sd          B.182 utt1:  My other son is just as happy as a bed bug. /

bc          A.183 utt1:  A clam. /
```

Sometimes (but not always) the speaker acknowledges the error afterwards.

```
sd          B.38 utt2: {C and } I suppose they all have the balloons. /

bc          A.39 utt1:  The air bags,  /
b          A.39 utt2: yeah. /

b^m          B.40 utt1:  The air bags,  /
b          B.40 utt2: yeah.  /
```

6.3 Answers

SWBD-DAMSL treats answers quite differently than DAMSL. First, where DAMSL has no subtyping of answers, SWBD-DAMSL answers are divided into 4 classes. Second, in order to speed up coding, we code the unmarked situation with a null label:

```
Answers-to-(pragmatic)-yes-no-questions

    affirmative answers
          ny            affirmative answers that are "yes" or a variant
          na            affirmative answers that are not "yes" or a variant
          ny/sd^e       affirmative answers that are "yes" and then an expansion

    negative answers
          nn            negative answers that are "no" or a variant
          ng            negative answers that are not "no" or a variant
          nn/sd^e       negative answers that are "no" and then an expansion

    other answers
          no     none-of-the-above (maybe, i don't know, etc)
          nd     disprefered response (well...)
          ^h     hold

Answers-to-non-yes-no-questions
   The immediate response to a non-yes-no question, (qw, qo, etc)
```

```
is *assumed* to be the answer unless it is marked with
'^h'  hold-before-answering.
```

6.3.1 Yes and No answers ny and nn

"ny" is only "yes", "yeah", "yep", "uh-huh", and such other variations on "yes".

We mark ny even if there's a filled pause or discourse marker along with the "yes". These are all ny's, counts from the first 18 conversations:

```
 17 yeah
  5 yes
  5 uh-huh
  3 {F uh}, yeah
  2 {F oh}, yeah,
  1 {F oh}, yes
  1 {D well}, yes
  1 yes {F uh,}
  1 yes, actually
  1 yeah, I do
  1 yep
```

nn is "no" and variations: Counts are from about 942 nn's from the first 755 conversations:

709 no (75%) 49 uh no (5%) 45 huh-uh (5%) 22 well no (2%) 19 oh no (2%) 16 um no (2%) 11 uh-huh (1%) 9 no uh (1%) 5 nope 3 uh actually no 2 yes 2 yeah 2 so no 2 probably not 2 but uh no 2 but no 2 actually no

*Coder's Heuristic

ny doesn't include "he is" or "he does". A.49.1 is *not* an ny, it's an na:

```
qy         B.48 utt1:  Is that the only pet that you have? /
```

```
na       A.49 utt1:  It is,  /
```

If the answer begins with "yes" and then *in the same slash-unit* expands on the yes, ^e can be added (i.e. ny^e) to mark a yes/no answer that has the expansion in the same slash unit:

```
qy          A.1 utt1:  Okay, {F um, }  Chuck, do you have any pets # there at you
```

```
ny^e          B.2 utt1:  # Yeah, I do. # /
```

6.3.2 na [a for 'affirmative']

An affirmative answer to a preceding y/n question that does not contain 'yes' or variations.

```
qy          B.16 utt2: do you have kids? /

na          A.17 utt1: I have three. /
```

Another example:

```
qy       A.67 utt1:  {C And } do [ they, + they ] just paper train it or some thin

na        B.68 utt1:  I guess. /
```

6.3.3 ng [g for neGative]

For negative answers to a preceding y/n question that does not contain 'no' or a variation.

```
qy          A.18 utt2: did you happen to see last night the special on
Channel Two with James Galway? /

ng          B.19 utt1:  We don't get Channel Two.  /
```

6.3.4. no [o for 'other' answer]

For responses to y/n questions that are neither affirmative responses ("yes" or "Indeed I do") nor negative responses ("no" or "I don't think so"). The most common case is "I don't know:

```
qy          A.15 utt2:  Do you think the jury should have a dollar figure
              for losing an arm, a dollar figure for losing different body parts? /
no         B.16 utt1:  I don't know.  /
```

6.3.5 ^e

The first statements by the same speaker after a yes or no response *to a question* have an sd^e, sv^e. These mark statements which are 'expansions' of the yes/no answer.

```
nn         B.56 utt2: no.  /
sd^e       B.56 utt3: [ I, + I ] live alone in an apartment,  /
```

We chose to mark *only* the first utterance after the yes/no answer, even though it will be often the case that utterances after the first one are also "expansions" of the yes/no.

^e can also be added to ny (i.e. ny^e) to mark a yes/no answer that has the expansion in the same slash unit:

```
   qy          A.1 utt1:  Okay, {F um, }  Chuck, do you have any pets # there at you

   ny^e          B.2 utt1:  # Yeah, I do. # /
```

^e is only added to the first utterance which contains the elaboration:

```
ny    B2.utt1. Yeah, /
sd^e  B2.utt2. we do have the death penalty here./
sd    B2.utt3. It’s not exercised very often,/
sd    B2.utt4. {C but} we do have it./
```

*Coder’s Heuristic

We are *not* marking expansions after answers that do not consist of ’yes’ (ny) or ’no’ (nn).

```
qy          A.22 utt2:  Do you ride a lot of rallies or a lot of those
        around there? /

ng          B.23 utt1:   Not so much.  /
sd          B.23 utt2: {F Uh, } I guess mostly I bike on my own.  /
```

6.3.6 nd "aNswer Dispreferred"

"Dispreferred" responses are marked ’nd’. These are pre-answer sequences of two specific types: answering negatively to a question that presupposes an affirmative answer or responding negatively to a question that presupposes an affirmative) often start with a hedge. This pre-answer sequence is marked "nd" (aNswer Disprefered). Yes-no questions generally presuppose an affirmative answer as do tag questions with a negative tag:

```
        You like Clinton, don’t you?
        Yes, I do.
```

Formal tag questions with an affirmative tag, on the other hand, presuppose a negative response:

```
        Question:                You don’t have a problem with that, do you?
        Preferred Response:      No.
                                 I don’t.
```

Where these patterns are contradicted by speakers, we may expect dispreferred markers, ’nd’ as in the following examples:

```
qy          A.63 utt1:  {F Um, } you kind of think it’s something else then? /

nd          B.64 utt1:  {D Well, } that’s what the environmentalists were
                claiming in this article.  /
```

```
qy          B.100 utt1:  Do you and your husband like to work in the yard?/

nd          A.101 utt1:  {F Oh, } {D well, } we like it once in a while
                  but not as often as we have to do it . /
```

(*Question: do we have examples responding to tag questions with ’nd’ in our database?)

If the dispreferred pre-answer sequence is transcribed in the same slash unit as the ’no’ answer, it is not coded. Rather, the answer itself is coded ’nn’ as shown in this example:

```
bf          B.66 utt1: Okay.  /
          B.66 utt2: {D So, } [ [ you, + you were out of s-, ] + you went
        to school ] for awhile and quit. Then  went back. /

nn         A.67 utt1: {D Well, } no.  /
```

Here B starts a "disprefered response" sequence, (Well...) but then A changes the question allowing B to answer "yes".

```
qy     A.5 utt1:    {D Well, } it should be used as a deterrent do you think? /

nd     B.6 utt1:  {D Well, } -/

qrr*   A.7 utt1:  {C Or } should it be used, {F uh, }  /
+*     A.7 utt2: [ a-, + ] to prevent further, # {F uh, } crime? #

sy     B.8 utt1:  # Yes,  /
```

Some additional examples are shown below:

```
b          A.1 utt1: Okay, {F uh, }  /
qw^t          A.1 utt2: could you tell me what you think contributes most to,
                {F uh, } air pollution? /

nd          B.2 utt1: {D Well, } it’s hard to say.  /
```

```
sd          A.203 utt1:  A lot of people say it doesn’t matter where they live if
                they have a nice house  /
%          A.203 utt2: [ {C and, } +

nd          B.204 utt1:  {D Well, }

sd          A.205 utt1:  {C but } ] I disagree with that,  /
%          A.205 utt2: I. -/

aa          B.206 utt1:  I do too,  /
```

```
qy          B.100 utt1:  Do you and your husband like to work in the yard?/

nd           A.101 utt1:  {F Oh, } {D well, } we like it once in a while
  but not as often as we have to do it . /

b          B.102 utt1:  Yeah. /
```

If the pre-answer sequence is transcribed in the same slash unit as the 'no' answer, it is not coded. Rather, the answer itself is coded 'nn' as shown in this example:

```
bf          B.66 utt1: Okay.  /
          B.66 utt2: {D So, } [ [ you, + you were out of s-, ] + you went
        to school ] for awhile and quit. Then  went back. /

nn         A.67 utt1: {D Well, } no.  /
```

In the following example, B.84 utt1 is NOT an 'nd' because it is not followed by a dispreferred answer:

```
b          B.82 utt1:  Yeah,  /
ba          B.82 utt2: I know.  /
sd          B.82 utt3:  [ [ They, + they, ] + they're ] just spoiled rotten,  /
%          B.82 utt4: [ {C but, } + {F uh, }

x          A.83 utt1:  .

b          B.84 utt1:  {C but, } ] no,  /
sd          B.84 utt2: [ I, + {F uh, } {F uh, } we ] love to eat out,  /
```

6.3.7 hold before answering ^h

This code started out as the DAMSL "Hold", but it drifted a bit; it now covers two kinds of phenomena that we would probably rather have separated out. The two are "true holds" (i.e. putting off the answer to a question), and "floor-holding holds" ("let's see", "what else now").

Type 1: If a question is not directly answered, but the response is nonetheless responsive in some way, it may be marked ^h (Hold).

If the response is itself a question, the question type is coded, followed by the ^h code, as shown in the following example (this is the standard DAMSL Hold):

```
qw    B.6 utt1:  {C And } what did you graduate in? /

sd    A.7 utt1:   I ju-, - /
qw^h  A.7 utt2: in what major or what year? /

b     B.8 utt1:  Yeah,  /
sd    B.8 utt2: major . /
```

While a ^h may be appled to questions as noted above, it may also be used as the complete marker for a slash-unit, as in the following examples:

```
qy^d          A.9 utt1:  {D Well, } like what? /

^h          B.10 utt1: {D Well, } let's see.
```

```
qw       A.1 utt2: could you tell me what you think contributes most to, {F
             uh, } air pollution? /

^h          B.2 utt1: {D Well, } it's hard to say.  /
sd          B.2 utt2: {E I mean, } while it's certainly the case that
        things like automobiles and factories, {F uh, } pollute a lot, {F uh, }
__________
qy          A.1 utt1:  Do you ever think that there's a crime that's just so heinous

^h          B.2 utt1: That's a good question.  /
```

The second use of "^h" is to mark things like "let's see" even if they don't directly follow a question (as in utt3 below):

```
qw          B.5 utt1: {D Well, } {D now, } {D so } if you were going to have a dinne

^h          @A.6 utt1: {F Um, } let's see,  /
sd          @A.6 utt2: {F uh, } I like seafood.  /
^h          @A.6 utt3: {F Uh, } let's see,
```

---

7. Other

7.1 quoted material ^q and (^q)

---

^q and **(^q)** are used to mark an utterance that has a direct quotation in it. (we code this because we suspect this may effect pitch and other prosodic features of the utterance).

If the quoted material is embedded in an utterance, the matrix utterance will be coded and the ^q code will be enclosed in parentheses. (so **sd(^q)** means a statement with a quotation in it, while **^q** means the entire slash-unit is a quote).

The illocutionary force of the utterance in which the quoted material is embedded will be coded, *not* the illocutionary force of the quoted material, as shown in the example below.

```
sd(^q)   B.32 utt1:   {C And } when the kids have kids come, {D you know, } s
        he's always saying, {D you know, } why do they have to be here,  /
^q      B.32 utt2: why can't they send them home,  /
^q      B.32 utt3: it's too noisy  /

sv      B.90 utt3: I think that's one of those things when we get to heaven
             we're going to ask God . /

ba      A.91 utt1:  I know. /

^q      B.92 utt1:  Why did you do it that way . /
```

7.2 hedge (h)

A hedge (h) is used to diminish the confidence or certainty with which the speaker makes a statement or answers a question. We code hedges only when they are in a single slash unit of their own (although of course there will be hedges in other utterances as well). Hedges may occur before the statement they diminish as well as after the statement. (The hedges we have been coding look very little like the sentence-internal hedges discussed in the semantics literature (Lakoff 1972, Kay 1987)).

The most common example of a single-slash-unit hedge seems to be "I don’t know." Here are some examples:

Hedge before statement:

```
br       A.19 utt1:  # The accuracy? #  /
h        A.19 utt2: I don’t know.  /
h        A.19 utt3: I don’t know.  /
sd       A.19 utt4: {C But, } I know there are a lot of things that can
              influence them  /
sv       A.19 utt5: {C and } I think that a person deserves a second chance
        with it or something  because  most things will stay in your system for a lo
```

Hedge after statement:

```
sv          A.103 utt1:  {C so, } [ I think, +  I think ]  that has helped a
        little bit,  /
h          A.103 utt2: I don’t know. /
```

```
+        A.25 utt1:  -- {D you know. }  Then you could lose out on a job when
              really you didn’t do anything. /

b        B.26 utt1:  Yeah. /

h       A.27 utt1:  # {C So } I don’t know. # /

sv       B.28 utt1:  # {C And } [ I, + I’m ] not # so sure they are that needed. /
```

```
nn          A.58 utt1:  #  {E I mean, } # no ,  /
sv          A.58 utt2: you probably know,  /
h          A.58 utt3: I don’t know. /
```

Hedges other than "I don’t know" will, again, only be coded ’h’ if they are contained in a single slash unit.

```
sd          A.45 utt1:  I have no interest in that,  /
sd          A.45 utt2:   [ I, + I ] don’t have interest of losing my ears,  /
h           A.45 utt3: let’s just put it that way,
```

```
b           A.41 utt1:  Yeah, /
h           A.41 utt2: I guess. /
```

Uncoded hedge (due to slash unit segmentation):

```
b          A.57 utt1:  Yeah,  /
sv          A.57 utt2: I think maybe they'd need to be  more knowledgeable though
than just your average Joe off the street --

b          B.58 utt1: uh-huh. /

+          A.59 utt1:  -- for something like that  because of the cultural
                differences.

b          B.60 utt1:  Right. /

+          A.61 utt1:  Things like that. /
```

The h code will NOT be used if the speaker uses "I don't know" to answer a question. In such a case, there is no hedge, as in the following:

```
qw^t          B.80 utt1:  How long is this going to go on, do you know? /

no^t          A.81 utt1:  I don't know. /
```

unless the speaker goes on to indicate knowledge as in the following, where "I don't know" is a hedge:

```
qy          A.35 utt4: however the question is is that making the difference. /

h          B.36 utt1:  {F Oh, } [  I, +  I  ] don't know.  /
sv          B.36 utt2: {C But } we have a lot of welfare programs  /
%          B.36 utt3: {C and } -- -/
```

7.3 How to use "+"

'+' is used to mark DAMSL's "Segment". SWBD-DAMSL has '+' because of our inability to alter the slash-unit segmentation of SWBD.

Coder's Heuristics

The following is a *wrong* use of +. Don't use a + if the same speaker finished their previous slash unit with a slash.

```
  sd         B.12 utt1:  {D Well, } it's, {F uh, } {D you know } they're just,
                         { F uh, } aggressive by nature -- /

  b          A.13 utt1:  Uh-huh. /

  +          B.14 utt1:  -- {C and, }  {F uh, } he's been neutered and declawed, # /
```

7.4 Double labels

Where two labels may apply, ';' is used to separate the two labels. The preferred label appears first, followed by the semi-colon, followed by the 'second-choice' code.

```
sv;sd          B.12 utt2: {C so, } {D you know, } really when you look at it,
                 they have full coverage,  /
```

We currently use only the first code for interlabeler reliability.

7.5 Transcription errors

Transcription changes are flagged by marking the affected utterance with "*", unless they are slash segmentation errors, in which case they are marked as "@". In general, we only mark transcription errors that directly affect the utterance coding.

7.5.1 Missing slash: one slash unit

If a single slash unit contains too much material (i.e., it should be broken down into more than one slash unit), the utterance is coded o@, as shown in the following examples:

```
o@    A.25 utt7:  Am I a pacifist, physical pacifist, I'm a Christian,  /
o@    A.25 utt11: {C but } I'm really not.  You know what I'm saying? /
```

7.5.2 slash units extend over utterances in error

When the slash unit extends over more than one utterance in error, and the first utterance can be coded in spite of the slash unit error, we code the first utterance 'code@' and the second utterance '+@' as shown in the two examples below:

```
aa@          A.27 utt1:  I tell you,
sv           B.28 utt1:  Boats are kind of expensive to maintain. /
+@           A.29 utt1:  {F Oh, } they are,  /
```

```
sd@           A.201 utt1: [ We  had, + we had ] two Siamese cats.
b           B.202 utt1:  Uh-huh. /
+@           A.203 utt1:  Different times. /
```

If the error extends over more than one slash unit, and no appropriate code is available due to the slash unit error, code the first utterance 'o@' and the second utterance '+@' as in the following example:

```
o@          B.42 utt1:  {D Well, } the,

%           A.43 utt1:  {D So, } /

+@            B.44 utt1:  other issue [ [ is, + is, ] + is ] [ how do you
           allow, + {F uh, } [ how, + how ] do you allow ] injustice.  Just
           like [ the, + the ] policeman [ in, +  in ] Los Angeles -
```

At the option of the coder, comments may be inserted regarding the coder's understanding of a correct code in light of the anticipated slash unit correction flagged by the '@' code.

7.5.3 Transcription errors in text

When transcription errors affect text only, the utterance is marked with * and a comment is inserted after the utterance, as in the following:

```
sv*          A.49 utt6: {C and } I know, like now  ((   ))  in China he did all
                  these terrible things,  / *[['Mao' not 'now']]
```

```
+*            B.82 utt1:  -- is ] if you look in the old test meat, and in the
                  numbers of places that, {F uh, } the Lord went out and just
                  simply struck down,  /  *[[old test meat = Old Testament]]
```

---

8. Bibliography

Allen, James and Mark Core. 1997. Draft of DAMSL: Dialog Act Markup in Several Layers. March 21, 1997

Bard, E., Sotillo, C., Anderson, A., and Taylor, M. (1995). The DCIEM map task corpus: Spontaneous dialogues under sleep deprivation and drug treatment. In Proc. ESCA-NATO Tutorial and Workshop on Speech under Stress, Lisbon.

Carletta, Jean. 1996. Assessing Agreement on Classification Tasks: The Kappa Statistic. Computational Linguistics 22, 249-254.

Carletta, Jean, Amy Isard, Stephen Isard, Jacqueline C. Kowtko, Gwyneth Doherty-Snwd and Anne H. Anderson. 1997. The Reliability of A Dialogue Structure Coding Scheme. Computational Linguistics 23.1 13-32.

Godfrey, J., Holliman, E., and McDaniel, J. (1992). SWITCHBOARD: Telephone speech corpus for research and development. Proc. ICASSP, pp. 517-520, San Francisco: IEEE.

Grosz, Barbara. 1978. Discourse analysis. In D. Walker, editor, "Understanding Spoken Language". 235-268. NY, NY: Elsevier, North-Holland.

Jefferson, Gail. 1984. Notes on a systematic deployment of the acknowledgement tokens 'yeah' and 'mm hm'. Papers in Linguistics 17.197-216

Kay, Paul. 1987. Linguistic competence and folk theories of language: Two English h in "Cultural Models in Language and Thought", edited by Dorothy Holland and Naomi Quinn, 67-77. Cambridge: Cambridge University Pres

Lakoff, George. 1972. Hedges: a study in meaning criteria and the logic of fuzzy c CLS 8, 183--228.

Meteer, Marie and Ann Taylor. 1995. Dysfluency Annotation Stylebook for the Switchboard Corpus

Weber, Elizabeth G. 1993. Varieties of Questions in English Conversation. Amsterdam: John Benjamins.

Sacks, Harvey, Emanuel A. Schegloff, and Gail Jefferson. 1974 A simplest Systematics for the organization of turn-taking for conversation. Language 50.4, 696-735.

Schegloff, Emanual. 1968 'Sequencing in conversational openings' American Anthropologist 70: 1075-1095

Schegloff, Emanual A. 1982. Discourse as an interactional achievement: Some uses of 'uh huh' and other things that come between sentences. In Analyzing Discourse: Text and Talk, edited by Deborah Tannen. Washington, D.C.: Georgetown University Press.

Schegloff, Emanual. & H. Sacks. 1973. Opening up closings. Semiotica, VIIII: 289-327

Schiffrin, Deborah. 1987. Discourse Markers. Cambridge: Cambridge University Press.

Yngve, Victor H. 1970. On getting a word in edgewise. Proceedings of Chicago Linguistics Society 6, 567-577.