7

Structured Coding for the Study of Language and Social Interaction

Martin D. Lampert
Holy Names College

Susan M. Ervin-Tripp
University of California at Berkeley

1. INTRODUCTION

The classification and labeling of natural events into discrete categories is a central part of most research in the social sciences (see Babbie, 1989). It allows investigators to identify and group similar instances of a phenomenon together for systematic study and is essential to any quantitative analysis. The process of classification and labeling is commonly referred to as "coding," and on the surface, coding appears to be a relatively simple task: (a) identify the information that you wish to recover, (b) select mnemonic abbreviations or numbers as codes to represent that information, and (c) do it—match codes to actual cases in your data base. In language research, this can involve the characterization of a number of linguistic and contextual features from phonology to event structure. For example, consider the following exchange:

Wife: Katie left her coat.
Husband: Uh oh.

If we were interested in syntax and morphology here, we could set up coding categories for noun, verb, tense, determiner, preposition, possessive, and so forth, and then assign codes for each category in such a way as to symbolically represent the original structure of each sentence. Du Bois and
Schuetze-Coburn provide a description of how to do this later in this volume, but as a simple illustration here, we might represent *Katie left her coat* as

Noun Verb+Past Possessive Noun

In addition to sentence structure, if we were further interested in the organization of verbal and gestural moves intended to control the behavior of others, and includes at present categories for some 66 facets of 17 dimensions considered to play a role in the organization of the control moves of children and adults in interaction. Because of its size and manageability, the CEC is a prime example for the issues and concerns encountered at each step of the coding process from construction to application.

2. THE CONSTRUCTION OF A CODING SYSTEM FOR LANGUAGE RESEARCH

Every coding project typically begins with some goal in mind. This can simply be to provide data with which to test a hypothesis or to explore possible dimensions of some phenomenon—as in the present situation, language. Once research objectives are in place, investigators can then begin to work on the construction of a coding system, a job that can proceed either in a top-down fashion, based on some theory, or from the bottom up, based on the raw data at hand.

In the former instance, investigators can and often do turn to a well-established theoretical framework as a guide to how to construct coding categories. In the case of language research, this means that a researcher may choose as a first step to select or develop a theory that explains how speech is organized and how certain variables are likely to influence language use. Once this is accomplished, the next step is then to figure out how to use the theory as a guide for what linguistic and paralinguistic phenomena to code (e.g., syntax, exchange structure, speech acts, prosody, etc.) and, more specifically, how to define these phenomena and break them up into coding categories.

The selection or development of a guiding theory is not easy. This requires not only a consideration of what theory, if any, suggests the best taxonomy for what the researcher wishes to study, but also a realization that whatever theory the researcher chooses is likely to have an influence on the kinds of analyses and results that are possible. In fact, two investigators can propose to study the same thing (e.g., speech acts) and code the same natural language set, yet if they begin with different theoretical orientations, they are likely to end up with different coding categories. They are likely to have different characterizations of the coded data and, in turn, different interpretations of the linguistic material studied. This point must always be kept in mind when dealing with theoretically derived coding categories.

On the other hand, of course, a coding system can be derived atheoretically; that is, established primarily to characterize what data are like, or altered when a theory turns out not to fit the data well. When researchers are not absolutely sure what categorical distinctions should be made, they can start with a few

---

1 The Control Exchange Code was originally developed in 1976 but has been revised in subsequent years. The original version of the code was developed by Susan Ervin-Tripp for family interaction videotapes collected in collaboration with David Gordon and Jenny Cook-Gumperz. There was a radical expansion of the code later by Julie Gerhardt and Lakshmi Savasri, who introduced a considerable deepening of the contextual information. Subsequent revisions and changes to make coding easier and more “user-friendly” and to adapt it to computer on-line coding were made by Martin Lampert working with classes of student coders. An additional section on overall request strategies was later borrowed from Shoshana Blum-Kulka, with some additional examples and definitions. Copies of the complete manual can be obtained from Susan Ervin-Tripp, Institute of Cognitive Studies, University of California, Berkeley, CA 94720, USA.
that this unit remain a constant throughout all coded texts and that, as with everything else, it be consistent with the investigator's research objectives.

For example, if an investigator is interested in the structure of discourse, the conversational turn may be a more appropriate unit of analysis than the sentence. On the other hand, if the interest is in syntax, the intonation unit or the clause probably would be more useful.

In the Control Exchange Code, our basic unit of analysis is the speech act with each act differentiated from the next by a shift in theme, purpose, or goal. In the following exchange, for example, we would divide the text into three separate segments:

Mother: Wash your hands then come down for dinner.
Child: Okay.

The mother's directives wash your hands and then come down for dinner would constitute the first two segments, because they specify two different actions, and the child's response would constitute the third.

Cases

Every segment of text defined by the basic unit of analysis, however, does not necessarily constitute a codable case. By "case," we mean specifically a segment that meets a set of coding prerequisites set down by the researcher. In many instances, a researcher may be interested in characterizing every segment in a transcript. For instance, someone interested in the structure of mother-child interaction might choose to code every turn, whereas someone curious about syntax may wish to code every clause.

However, if interest is on only a subset of exchanges, utterances, or acts, coding every segment in a text may not only be unnecessary but in some instances counterproductive. In the CEC, because our main objective is to study only control-oriented speech, we select for coding only those speech acts intended to change the behavior of someone else, such as a request for goods or services. We do identify how assertions, greetings, explanations, and other noncontrolling speech serve as support for or a reply to a control act; however, these acts do not receive separate coding records of their own. For instance, we would establish a separate record for each of the mother's directives in the example above, but not for the child's response. The child's okay would not be considered a case for coding purposes; however, in the coding records for the mother's directives, we would indicate that the intended addressee did agree to comply. In this respect, the CEC is radically different from coding systems that identify the function of every utterance.

Restricting what qualifies as a case has two advantages. First, it allows the researcher to design the coding system specifically around that aspect of
language that is of interest, and second, it provides for a more productive use of
time. If a coding system in fact makes many distinctions, the amount of
time necessary to code every utterance in a text could be enormous, with the
payoff for the additional work minimal, especially if only a subset of cases will
ever be analyzed.

On the other hand, coding only select segments also has its disadvantages.
First, it is single-purposed in that it allows only for a characterization of the
cases sampled and not the full data set. And second, when only some
segments are analyzed, the potential exists to overlook important dynamics in
the overall interaction. Clearly, there is a need to weigh the short-term
efficiency of coding only parts of a transcript against the long-term benefits of
having fully coded texts.

2.2. Description of the Data

Once the decision has been made as to what cases to code, the next step is how
to describe them. This typically involves setting up a record for each case that
contains coded information placed in some systematic order. The coding
system that one adopts generally lays out the form of this record by outlining
three levels of description that we discuss in this section: (a) topics, the actual
dimensions that are used to describe cases, (b) categories, the values that are
assigned to the individual topics, and (c) codes, the abbreviations associated
with categories that make up the coding record itself.

Topics

A coding system generally begins with the development of a series of
coding topics. Some topics may be set up strictly for bookkeeping purposes,
for example, to provide the line number of the coded utterance in the original
transcript, the location of the utterance on the tape, or other identification
information. Having coding records and their corresponding transcript lines
begin with the same identification number is often useful for easily separating
the two or sorting them together for various analytic purposes. For
bookkeeping purposes, one could also include as a topic a numerical index to
link related cases, such as repetitions, elaborations, and question-answer pairs.

Most topics in a coding system, however, are generally suggested by
theoretical orientation and research goals. The Control Exchange Code, in
particular, was designed to study the relationships between certain contextual
variables and the types of control acts used by parents and their children.²

²In a coding system designed to classify the functions of every line of dialogue, topics would most
likely include a richer list of exchange features, speech acts, and activity types that those identified
under the CEC. For example, in a more comprehensive coding system designed to study classroom
interaction (Ervin-Tripp, 1988b), we identified exchange features for every utterance, including
whether a turn initiated an exchange, maintained it with new information, questions, and back-
channeling, or terminated it altogether. Ninio and Wheeler (1984) provided a similar system for
the line-by-line functioning of mother-child interaction.

In the CEC, we did not examine noninstrumental acts or the overall structure of an interaction
as these were not of primary concern. We did, however, capture certain exchange features with
respect to the onset of an instrumental move—for example, the activity context, and whether the
move initiated or continued as exchange.
the construction of a control act. The first set of topics identifies the context in which a control move occurs. These include not only topics for identifying participants and events but also intent and the nature of the situational context and discourse leading up to the control move. The topics in this group are as follows:

**Speaker:** The individual who initiates the control move.

**Addressee:** The individual or group intended to act (or not to act) as a result of the control move.

**Event:** The activity context in which the control move occurs.

**Timing:** The level of engagement between the speaker and the hearer at the time of the control move.

**Repeated Tries:** The relationship between the current move and earlier related ones that may or may not have been successful.

**Cost:** The likelihood of the speaker being rebuffed or ignored as a result of making the control move.

**Purpose:** The reason for or underlying intent of the control move.

The second set focuses on the characteristics of the control move itself and of the supporting or tactical moves that Garvey (1975) identified as being within the domain of a request. These include topics to describe the linguistic form of the move as well as supporting linguistic and paralinguistic devices used to increase the move’s chances of being effective. This group includes:

**Basic Form:** The sentence structure of the head control move.

**Explicitness:** The extent to which the action, goal, and actor of the desired move are expressed or have to be inferred.

**Verbal Attention Forms:** The linguistic devices used to get an addressee to pay attention to the control move.

**Adjuncts:** The reasons, justifications, threats, and mitigators used to increase the likelihood of compliance with the control move.

**Gestures:** The facial expressions, postural shifts, and hand motions used either to direct an addressee’s attention to the control move or to emphasize its imperativeness.

**Vocal Mode:** Shifts in register or pitch intended to get an addressee’s attention or qualify or emphasize the move.

The third and final set includes topics that describe the effect of the control act on the intended addressee. The topics here include:

**Verbal Reply:** The verbal response of the addressee to the control move, expressing either a willingness or unwillingness to comply and why.

**Behavioral Compliance:** The level of success of the control move in getting an addressee to act.

All of these topics were incorporated into the CEC system so that specific empirical questions could be addressed. These included whether there were age changes in children’s tactics in getting attention, persuading addressees to act, and remedying failure; and whether relative age or other characteristics of the addressee affected forms used and the probability of success. A series of articles has examined these and other findings from the coding of family interaction (Ervin-Tripp, 1982; Ervin-Tripp & Gordon, 1986; Ervin-Tripp, Guo, & Lampert, 1990; Ervin-Tripp, O’Connor, & Rosenberg, 1984; Gordon & Ervin-Tripp, 1984).

**Topic Fields**

In the coding record, it is usual to reserve a space at a specific location and of a fixed length to enter information on each topic. We refer to these reserved spaces as **topic fields**.

In many instances, a topic will have just one field set aside for it. To flesh out a topic, however, it is often useful to make finer distinctions in the form of subtopics. A topic, therefore, can have just one field associated with it or serve as an umbrella for several interrelated fields, depending on the researcher’s needs. A researcher may choose to use more than one field to provide more comprehensive information on a topic. In some systems, for instance, the only space needed under **Speaker** might be a field for the speaker’s name. In the CEC, however, we also include fields for the speaker’s age, gender, and role-playing in the interaction because we see these variables as likely to influence the form of a control act as well as the compliance of addressees.

A researcher may also employ multiple topic fields to provide not only a more comprehensive description but also a richer characterization. For instance, while developing the **Event** topic for the CEC, we found that it was common for children to be distracted temporarily from one activity to engage in another or to embed one activity in another as a kind of aside. At these points we could tell that one was more foregrounded with respect to the control act but that the other was still in a way there, for the children picked up where they left off. Accordingly, we included fields to indicate a foregrounded as well as a backgrounded activity if both existed.
A final use of multiple fields is for simplifying the business of coding by breaking up a single topic, which may require a rather complicated coding decision, into less complicated subtopics that individually require fewer decisions to code effectively. This last consideration is discussed more fully in the section on implementation.

Coding Categories

Once topics have been outlined, the next step in the construction process is to lay out the categories that will be assigned to each topic field. One way to achieve this goal is to adopt the top-down approach of applying an already existent or theoretically derived classification system to ones data. For example, if we wished to code the speech act performed by every utterance in a text, we could adopt Searle’s (1976) speech act taxonomy and classify utterances with some success into the following five categories:

Representatives: Speech that commits a speaker to the truth of the expressed proposition (e.g., assertions, suggestions, conclusions).

Directives: Speech where a speaker attempts to get an addressee to act (e.g., requests, prohibitions, questions).

Commissives: Speech that commits a speaker to some future action (e.g., promises, offers, threats).

Expressives: Speech that reveals a speaker’s current psychological state (e.g., apologies, greetings, thanks).

Declarations: Speech that immediately changes a current state of affairs (e.g., christenings, firings).

In adopting this taxonomy, however, we must recognize that we are assuming that it will allow for the kinds of distinctions that we wish to make. This may not always be the case.

Frequently, especially if we are interested in only a subclass of language or have a novel perspective on language use, we cannot always use another investigator’s coding system. In the CEC, for example, we study a subclass of speech acts, which would suggest that we could have used Searle’s or a similar taxonomy to classify control moves. Searle’s system, however, would have been inappropriate here for it suggests categories that we would never use to classify control acts, such as apologies and christenings. Furthermore, based on Searle’s definitions, what we have termed a control act would always be coded as a directive (i.e., language where the speaker attempts to get the addressee to act). Yet, it is possible to see acts labeled as representatives and commissives as intended to change the behavior of an addressee in some fashion. For example, an offer of goods, which would be classified as a commissive in Searle’s system, could be viewed as a request for an addressee to act, that is, to accept the goods provided.

When an appropriate classification system is not available, the alternative is to adopt the bottom-up approach, that is, to begin with selected transcripts and identify categories that appear to occur naturally in the data. This was the method that we adopted for the CEC. Under the topic of Purpose, for example, we developed a field labeled Type of Act, with empirically derived categories to capture speaker intent. We began with obvious categories, such as request and prohibition, but added categories for other types of speech that appeared to require an addressee to acquiesce or comply. In all, we developed the following six coding categories:

Directives or Positive Requests: Utterances that require that an addressee act to provide either goods or services.

Prohibitions or Negative Requests: Utterances that require that an addressee stop or avoid a line of action.

Permissions or Allowances: Utterances that either request from or grant to an addressee permission to obtain goods or services.

Intentions: Utterances that commit the speaker to a line of action that an addressee is expected either to facilitate or at least not block.

Claims: Utterances that require an addressee to recognize a speaker’s right to certain goods, activities, or services.

Offers: Utterances that invite an addressee to accept goods or services.

Notice here that the CEC’s categories encompass a much narrower band of language use than Searle’s speech act system as a result of our focus on just control acts. Also, notice that the CEC divides up control acts in a manner different than that suggested by Searle’s directive category. Like Searle, we identify requests and prohibitions as language intended to get others to modify their behavior; however, we also pull in intentions, claims, and offers, which, if we had used Searle’s categories and definitions, would not be considered control acts at all, let alone coded for type. In short, by looking at our data first, we were able to see the kinds of things that speakers attempt to get addressees to do (e.g., accept offers, respect ownership, allow a stated intention), and accordingly, we were able to construct categories that were appropriate for our research needs.

Coding categories, then, can be suggested by theory, borrowed from another investigator, or empirically derived. Irrespective of the method used, however, we would like to suggest three rules of thumb. First, coding
categories should follow directly from research goals and should not include distinctions that have little or no utility for future research objectives. That is, investigators should not employ another’s coding scheme simply because it is accessible and there may already exist available texts coded with this system. An earlier coding system, although appearing to capture the dimensions an investigator wishes to study, may in fact be inappropriate for a research objective other than the one for which it was initially created.

In the case of *Type of Act* in the CEC, the six basic categories were selected after a consideration of how a range of different types of speech acts were likely to affect the behavior of an addressee. Accordingly, we included categories for types of speech that could be seen as intended to get an addressee either to act (or not to act) or to facilitate (or at least not block) the speaker’s current plans.

A second rule of thumb is that categories within a field should be clearly defined so that it is clear not only what each represents but also how each one differs from the rest. In short, the categories should be defined, if possible, so that they are mutually exclusive. This is not to say, however, that a case cannot be, and should not be, coded into more than one category. In fact, given the multifunctionality pervading language, double coding may not only be useful, but desirable.

For example, consider *Do your own work* said by a teacher to a student caught looking at a neighbor’s exam during a test. At one level, in the CEC system, this utterance could be coded as a directive, because the teacher is directing the student to do something (i.e., work on his own exam). On another level, however, the statement can also be heard as a prohibition because the teacher is also trying to prevent the student from doing something (i.e., from copying from a neighbor).

For cases like this, there are two mechanical conventions available for the double coding of data. The first is to create not one but two coding fields to allow for a primary and a secondary coding. As mentioned earlier, this solution was adopted in the CEC system to deal with event codes. This solution could have also been adopted to handle the coding of acts with multiple functions. By opening up a second field for type of act, it would have been possible to code *Do your own work* as both a directive and a prohibition. Notice, however, that this particular coding solution requires a reorganization at the topical level.

An alternative solution is to set up a series of “joint” coding categories, ones that represent the co-occurrence of two possible outcomes, such as an utterance that may serve as both a directive and a prohibition, as in the case of *Do your own work*. When constructing a joint category, however, care needs to be taken not to violate the rule of mutual exclusivity. This is accomplished by making clear in the definition of a joint category that it applies only to those cases that fit the definition of a first category and that of a second. If only one definition applies, then the appropriate stand-alone category (e.g., *directive*), and not the joint category (e.g., *directive-prohibition*) is used. In the CEC, we employed this second solution to capture control acts that could serve as both a directive and a prohibition (e.g., *Do your own work*) as well as those that could serve as a directive and an intention (e.g., *Let’s go to the movies*).

The final rule of thumb is that all categories within a field should be not only mutually exclusive, but also exhaustive. That is, within each field, the coding categories should be extensive enough to be able to provide a classification for every foreseeable case, even if this requires including a *miscellaneous* or *other* category.

Some researchers may have theoretical reasons to force cases into a limited set of categories. However, one cannot always predict what all possible cases might be like. The hope is that through careful consideration of how cases may differ for a particular topic area, it would be possible to isolate all possible coding categories. Work by other investigators and various theories on how language is organized can provide clues here. However, for a newly constructed topic field there is likely to be a case that just does not quite fit into any category.

The solution to this problem in the short run is to create an *other* category. The value of such a category is that it provides not only a place for uncodable cases but also an avenue for exploring how to revise and expand the coding system.

When an *other* category begins to catch a substantial number of cases, the coding field as originally conceptualized needs to be modified in some fashion. The ways in which *other* cases cluster may suggest that either a new coding category should be added or that the definitions and distinctions between the

---

3 Some may argue that this is an impossible task, requiring the identification of core features that may or may not, in fact, be representative of all exemplars of a category. This criticism, however, overlooks the fact that, for descriptive and analytic purposes, it is in the researcher’s best interest to use care and objectivity when defining the criteria for distinguishing categories, so that cases can be assigned to categories effortlessly and unambiguously.

4 The reader may appreciate at this point that based on the six original coding categories under type of act, 30 different joint coding categories would have been possible. However, we did not choose to provide codes explicitly for the remaining 28 possibilities for empirical reasons. After having coded some 20 transcripts with the CEC, we encountered few utterances codable as serving dual functions, and those that have been encountered are of the two types already identified. It is primarily because of the infrequency of dual function control moves that we did not opt for the two-field coding solution (i.e., the second field would have been left uncoded most of the time) and why only the two mentioned here are formally defined in the CEC coding manual. Because of the recognizable problem in cases like this, some code designers do provide preference hierarchies, instructing coders how to make a forced choice between the two categories in joint cases. We did not do this.
original categories should be revised in order to incorporate the unclassifiable cases.

The inclusion of an *other* category should not be seen as a weakness in an ongoing coding project, but in fact a strength. Such a category helps to guide revision of the coding system as a whole and to make clear that the development of a coding system is not a static one-time operation that occurs before the coding of the first case. Rather, the construction of a coding system should be seen as a dynamic and ongoing process that benefits at the start from original theories and research goals, yet has enough flexibility so that later revisions can be made to capture unexpected and interesting distinctions.

**Coding Labels**

The final consideration in the construction process is the selection of codes or coding labels. A coding label is the abbreviation or tag added to the data to represent a coding category. In principle, they can consist of any alphabetic or numeric characters, but errors in data entry are less frequent when they have some mnemonic resemblance to the name of the coding categories they represent (e.g., *D* for Directive, *I* for Intention, etc.) than when they are arbitrary numbers (e.g., *1* for Directive, *2* for Intention, etc.).

For minimizing error and easing data analysis, it is preferable for all the labels within a particular field to be of the same length, that is, the same number of characters. Computer analysis programs such as the *Statistical Package for the Social Sciences* (SPSS) require the researcher to indicate the number of characters for each field and expect it to conform to the specification. If a code label is said to be two characters long, it must fit within two characters or else the data will be read incorrectly. Although spaces can be used to pad shorter values to make them equal, this increases the possibility of error, as coders may not place spaces uniformly and some software packages may not interpret "D" and "D" as being equivalent.

If labels differ in length within a field, they can be modified by global substitution prior to the actual analysis to conform to the requirements of the software package. The main thing, however, is that codes should be systematically differentiable and easy to check for errors, and both of these goals are facilitated when field lengths are kept constant.

2.3. Some Limitations

Every coding system has limitations that become more apparent as one attempts to reduce the complexities of recordings and transcripts to coding categories. Even after initially designing and periodically adjusting the CEC, we continued to be troubled by the fact that we had forced ourselves to deal with only one level of a highly complex layering of functions. A narrative, for example, can be embedded within and support a control act, and a control act can be embedded within and support a larger scheme or strategy. Labov and Fanshel (1977) discuss this layering most fully; we did not deal with it at all except at the Speech Event level. Not surprisingly, we found that coders often differed in choosing the level at which to interpret an activity or event.

Another limitation we uncovered arose from our choice for unit of analysis. Because we focused on and evaluated only one speech act at a time, we developed a kind of myopia to certain features in the dynamics of a conversation. For example, in one of the tapes we had used many times for training coders in the CEC, we noticed that one child always dominated another, in every context. Her preference always won, her topics won, her choices of play won. She was, in fact, a few months older than the other child; her strategies were subtler. We had looked in the text only at the dynamics of each act, one at a time, not at their relation or organization. We did not get the overall picture of the relationship. Our failure to see these dynamics even though they were pertinent to the issues in our research is evidence of how narrow one's focus can become in coding. This limitation may be impossible to remedy in the framework of a coding study, because it depends on constantly reframing one's perspective on the data. Coding is based on maintaining a consistent and shared perspective. This is why qualitative study of the same texts is always necessary as a complement to the coding process.

3. THE IMPLEMENTATION OF A CODING SYSTEM

3.1. Coding as Language Planning

Once the basic outline for a coding system is in place, the next step is to put the system to work—that is, to train coders on how to use the system to describe a set of texts. But what is this process like? How can it be organized to the researcher's greatest advantage?

The learning of a code in essence can be compared to the learning of a vocabulary in a second language: Both require learners to develop a sense of prototypical cases as well as categorical boundaries for newly acquired terms. As in second language acquisition, the easiest coding categories for a learner to acquire are those that involve a familiar concept such as adult or past tense. This sort of category is comparable to learning a new word in a second language that has an equivalent in one's first language. However, the *faux amis* of translation reminds us that analytic codes may restrict meaning more than is conventional in ordinary language. For instance, the word *nom* in French in most contexts means "name," but in the context of filling out official forms it means just "family name." Many an erasure testifies to this problem for English speakers learning French in France. An example from
transcription is the use of the question mark for rising intonation alone, not for grammatical interrogation, as in *What do you want*? versus *What do you want?* Some code-builders avoid such forms altogether to prevent interference from conventional usage.

Safeguarding against interference from differences in range of reference requires prudence and foresight, or the luck to discover coder confusion early enough to change the code. Ideally, codes should draw on concepts that are natural, already familiar from practical experience, or very easy to learn. Thus, the process of coding can lead coders to identify "natural" behavior units. When codes create theoretical, arbitrary, artificial units and artificial coding concepts, there is a high risk both that the concept will be short-lived in the coders' minds and that it will be short-lived in the history of the research field.

When presented with categories named by a researcher, coders become in essence newcomers to a new group who must learn the group's jargon. In natural language learning, we rely on context to teach; in code instruction, we cannot do this, and instead must be as explicit as possible. Extensive training against modeled coding supplies the equivalent of context. The problem with training by coding of natural texts is that due to natural frequencies some categories are not trained at all; others are very well taught. Experience with children's learning of vocabulary tells us that variety of contexts of exposure is crucial to new learning. So for every category, a range of examples needs to be available.

Extensive training and examples, however, do not guarantee perfect coding. Even after being thoroughly trained, coders may still apply a category in a manner different from that laid out in the coding system because of a "drift" in meaning. What is likely to happen is that the conceptual correlates for coding terms may tend to move toward a coder's natural category system. As codes are defined verbally, variations in the coders' sense of word meanings can therefore lead their personal definitions for a coding category to drift apart from one another as well as away from the original intended definition.

An additional problem is that coders from different backgrounds often bring with them different presuppositions and concepts that can influence their decision making, and subsequently, their coding judgments. For example, we discovered that coders would often disagree about whether a request for goods involved a high or low cost to a speaker because of different familial and cultural views about who has rights to such things as food and toys and therefore can ask for these things directly without the fear of criticism. We further observed examples of such bias in a project in which we coded for conversational humor. Cultural background, gender, political viewpoint can all influence the identification of the basis of humor, and we found that recent immigrants did not recognize idioms or cultural allusions at all.

In short, coders typically have implicit ideas about how certain phenomena correlate with one another, and consequently, they may code topics—albeit at times unintentionally—so that these underlying assumptions are corroborated. This is why constant communication and retraining is needed to keep meanings calibrated and mutually acceptable. In addition, the communication helps clarify ambiguities in the researcher's thinking and helps in code revisions to add examples, clarify definitions, and even identify new categories that are needed. A regular "Round Robin" with paired duplicate coding is one way that coding projects do this; it plays the same role as talking about the same visible referent does in language learning.

We can see then that there is a paradox in the coding process: that is, even a carefully outlined system with clearly defined topics and categories does not guarantee that coders will be able to use the system to make valid and reliable coding decisions. In the CEC, for example, we define directives under *Type of Act* as simply speech that requires or requests an addressee to provide goods or services. This definition, however, does not say anything about what a coder should and should not consider when making a decision about whether an utterance should be classified as a directive or not. Based on the definition alone, we have no clue, for example, as to whether a statement such as *Katie left her coat* should be seen as an assertion and left uncoded or seen as a subtle hint to act and coded as a directive.

The second major task in the coding process, therefore, is not simply to write better definitions—as better definitions are not likely to solve the problems cited here—but rather to figure out how to present and teach the coding system so that coders can learn its language and use it to assign cases smoothly and objectively to categories. In this section, we provide suggestions that can be used to meet these goals by looking at the organization of topics, the presentation of coding categories, and the mechanics of data entry.

3.2. The Organization of Topics

The researchers in charge of a coding project may do their own coding; in which case, they may already have a good idea of what distinctions they wish to make within each topic area. Frequently, however, individuals not involved in the planning and development of the original coding system are solicited to do coding work, and they have to be provided with an outline for doing coding and taught how to use it.

**Topic Definitions**

The initial problem, as already highlighted, is that newcomers often come to the task of coding with their own definitions and ideas about such things as syntax, speech acts, turns, gestures, and other language-related phenomena.
These prior definitions could conceivably facilitate the coding process to the extent that the new coders’ definitions match those employed by the coding system. To the extent that they do not, however, old definitions could easily interfere with and cause confusion over what the researcher actually wants coded under a given topic and could lead to invalid coding decisions.

A standard example in the study of requests involves a phenomenon known as *prerequests*. These are strategic, preparatory moves intended to get an addressee to orient toward a desired object, or to gather information on the likelihood of compliance, prior to the actual delivery of a request. In the following sequence of turns, the problem is how to consider the first two turns. They can be seen either as preparatory moves before the head request in the third turn, or they could be considered part of a series of repeated requests, increasing in explicitness when the first moves fail.

1. I don’t have enough stickers.
2. Are you all done? (Nods)
3. Could I have some?

Levinson (1983) has argued that the genesis of conventional indirect requests might be exchanges in which the preferred outcome was a compliance by the addressee who infers what is desired, leaving the requestor successful without having taken the risks of refusal of an explicit, on-record request. Therefore, in any particular exchange we cannot know for sure, especially if the speaker is a young child, whether the moves were essentially a strategic sequence or a sequence of repeated tries. Undoubtedly, coders are affected by their beliefs about age and capacity in this judgment and must be given rules for making these distinctions in some standardized way.

Accordingly, a coding system must not only lay out what each topic involves from the researcher’s perspective, but also how its current conceptualization differs from some dictionary, folk, or previous theoretical definition. When we define the topic of *Purpose* in the CEC, for example, we spell out exactly what coders must and must not consider when making judgments here:

Control acts are attempts to get another person to act, often with strong social purposes as well. In terms of the action component, we can conceive of a variety of goals in terms of changes in location of goods, activities, and states, varying beneficiaries, and different levels of the conscious attention of the speaker. In common sense language, these variations are coded into terms such as order, offer, prohibition, and so on. For instance, an offer has the addressee as beneficiary.

The point of this topic is to determine what the speaker wants to accomplish through each control act. The first judgment is to decide whether the desired outcome of the act will primarily involve action by the hearer or by the speaker. If the action is by the speaker, then we are talking about offers, intentions, and soliciting permission. If the expected action is by the hearer, we are talking about directives, prohibitions, claims, and permission-giving. The analysis of purpose often can go deeper. We are looking at purpose just behind the surface forms, not in terms of deeper unconscious goals. Try to decide what the goal is for the speaker. This will help you see whether the speaker’s focus is on the goal, on the means, or on what is annoying, that is, on what is to be changed.

Our comments here also apply to the definition of coding categories within each topic area, and we readdress this issue in the section on categories below.

**Topic Order**

Clear-cut definitions can reduce coding problems, but they do not eliminate them. Another difficulty stems from the effect that earlier coding can have on later work. Coders may at times code a particular topic field in line with an earlier one because they (a) rely on the earlier field to help them in making later coding decisions, (b) have implicit notions of how the two fields should be related, (c) hope to confirm a theory, or (d) hope to disconfirm one. For example, if a coder identifies two statements as successive control acts, he or she would be unlikely to code the first statement as a preparatory move for the second, even if the two moves could be seen in this way. Similarly, what coders identify as a Foregrounded Event is likely to influence their view of the power structure, and later on their judgments of Cost.

Given all this, how topics are arranged within a coding system and assigned to coders becomes a critical issue, especially if the same individual is expected to code all topics for each individual case in a text. Left unchecked, the effect of earlier on later coding could invalidate the coding system.

As discussed by Bloom (this volume), this problem can be minimized by having individuals code different topics. Having a different coder for every topic, however, may not be particularly efficient, especially if the coding project has only a few coders and many fields. Another solution is to have the same individual code each topic but to do so on a separate day without reference to any earlier coding. This, however, has the disadvantage of increasing the amount of time it takes to code a single text.

In the CEC, we employ a combination of both solutions. For any given transcript, we have different people code different topics, but this is not a one person–one topic arrangement. Coding assignments are done primarily by splitting the system in thirds according to the three larger topic areas identified earlier. Because the system was designed mainly to study how context and intent are related to the form of a control act and how form in turn is related to compliance, we felt it wise to have different individuals code the context,
form, and compliance variables in order to avoid any bias toward or against a particular hypothesis.

For each assigned block of topics, we also instruct coders to make several passes through a text and to code a different topic on each pass. The reason for this is to minimize the effect that an earlier coding decision in one topic area for a case could have on the coding decision in a subsequent topic area. We have developed computer software to facilitate the coding of one topic at a time, which we describe in more detail in the section on data entry below.

**Field Order**

The concerns that apply to the arrangement of topics also apply to the organization of fields within each topic area: namely, the concern that the coding of one field may influence the coding of another. Depending on what the topic fields reflect, however, the previous solutions could turn out to be unnecessary and, in some cases, even problematic.

First, separate coders and separate passes are not necessary when it is clear that decisions required for two fields are completely independent of one another and can be objectively made. For instance, under Speaker in the CEC, a decision concerning the coding of a speaker’s age is unlikely to influence a decision concerning the coding of the speaker's gender. These two fields can be safely coded together.

Second, separate coding may in fact be undesirable when the coding of one field is dependent on the coding of another, such as in two fields used to specify the type and location of a linguistic feature.

For example, in the CEC, we code attention forms that co-occur with a main control act. These can be of various types (vocatives, epithets, exclamations, turn takers, etc.) and can appear in different locations relative to the act itself. There can even be more than one attention form. If an attention form is identified, its type and location need to be coded at the same time. Otherwise, it is possible that in a first pass the attention form may be caught and its type coded, but in a second pass overlooked with its location left with a null code (a code that indicates that location is not specified because there is no attention form to identify). In sum, these two fields must be coded together because they are mutually dependent on one another (i.e., if one field has a non-null code, the other must have one as well).

Third, separate coding may also be undesirable when one field constitutes a subclass of another. In this instance, not only must each field be coded in the same pass, but in a specific order. Employing fields that are subclasses of one another, in fact, can be quite helpful in making category decisions. As mentioned earlier, when a coding decision may be rather complicated, it is sometimes helpful to break the decision making process into two steps: a first step where a primary coding decision is made, and a second where a subsequent and finer distinction is worked out.

In the CEC, in fact, we employ two separate and hierarchically arranged fields to code adjuncts (i.e., statements that support or justify a control act). The first is used to provide a major type category (i.e., “reason,” “threat,” “history,” “clarification,” etc.), and the second is used to make a finer distinction depending on what was coded in the first field. For example, if a control move involves an adjunct (some do not), and the adjunct is coded as a reason in the first field, then the system allows for one of several types of reasons to be specified in the second (i.e., “external state,” “internal bias,” “situational norm,” etc.).

The use of subclasses as fields is a technique that can also be employed at the topic level to facilitate the coding of categories. Other techniques involving the coding categories themselves can also be used, and we now turn to these.

### 3.3. The Presentation of Coding Categories

When setting up a topic fields, it may appear sufficient to provide a clear definition of each coding category and examples of illustrative cases. However, as with topic definitions, seemingly well-defined categories do not take care of the second language problem.

First, coders may use inappropriate criteria for deciding whether or not a case fits a particular category definition. Second, although definitions may suggest mutually exclusive categories, real world cases may still be difficult to classify, especially if whether or not a case can be classified into a particular category depends on making inferences from overt behavior and speech that are not part of the category’s definition. And third, even with clear-cut definitions, there may still be fuzzy cases that can fall into more than one category.

The overriding problem here is the possibility that coders will classify cases into a particular category based on the category’s probability of occurring within a given context or with an identifiable linguistic form that has nothing to do with deciding whether or not the category should be used. This problem would not be so serious—although it is a serious one—if not for the fact that analysis often involves relating topics to one another. To the extent that relationships between topics pertaining to context, form, and other language-related dimensions are to be studied, such coding could lead to spurious correlations.

For example, we might end up with a perfect correlation between form and purpose if all declarative sentences were coded as assertions and all imperatives and interrogatives as directives. Statements such as The window is
open. It’s naptime, and Katie left her coat would all be coded as assertions although under some circumstances they could be interpreted as directives.

The point is that a coding system must make clear what is and is not to be considered—or at least what criteria should be given primacy—when deciding in which category to place a case. In short, the coding categories must be operationalized. That is, guidelines must be set for what coders must observe in the natural language data before they select a conceptually defined category. Some guidelines can be anticipated; others, however, can only be found in the course of actual code use.

As a result of coding experiences, for example, under Type of Act, the CEC does not simply define a directive as speech that requires or requests an action from an addressee, but rather also instructs coders that they should not be distracted by the surface form and that they should consider factors that indicate (a) that the speaker is in need of certain goods or services, (b) that the hearer can be expected and/or is expected to meet that need, and (c) that the hearer does not have the option to refuse. Under this definition, Have an orange looks like a directive but might be considered an offer, and I’ve finished to a waiter becomes a directive that does not look like one.

Stipulating that coders should look for specific observables in a text before they select a category, however, does not guarantee that they will do so effectively. One solution to this dilemma is to outline for each topic a step-by-step procedure for making a classification. This can be guided by category definitions and take the form of a hierarchical flowchart with each level of the chart representing a different yes–no decision as to whether something is true about a case or not. To be easy to use, a coding chart should follow the normal path of coders using code decisions, and the final level should contain the individual coding categories. Figure 7.1 provides a possible flowchart for coding Type of Act in the CEC.5

One obvious flaw in this hierarchical arrangement is that some cases may be viewable in several different ways. As we have seen, for example, the utterance Do your own work can be interpreted as a directive and a prohibition. Under these circumstances, what do we do?

One solution, of course, is to code only one interpretation. However, some researchers may wish to represent this duality. Other solutions are ones we have already suggested. First, separate coding fields could be open for each possible interpretation, with each following a different path down the hierarchical flowchart. And second, joint coding categories could be created, and these could be configured into the hierarchical arrangement.

---

5A hierarchical arrangement for making coding decisions, of course, can be built into the code itself by creating a separate field for each coding decision as was illustrated for the coding of adjuncts in the preceding section.
hierarchical flowchart. And second, joint coding categories could be created, and these could be configured into the hierarchical arrangement.

Even with operational definitions and flowcharts, however, a coding manual is not complete. To guide them along, coders not only need definitions but also examples of prototypical and boundary cases to give them a sense of what they are looking for with respect to a particular category. Because instances occurring at boundaries, or points of potential ambiguity between categories, may be rare in natural texts, the code or the training materials need to oversample at boundaries where coders are most likely to disagree. Along with good definitions and flowcharts, a broad range of examples can also provide coders with the tools they need to code successfully. As an illustration, Table 7.1 contains the CEC manual page for Type of Act, complete with definitions and examples.

| TABLE 7.1 |
| Manual Page from the Control Exchange Code for Type of Act |

**Topic 7: Purpose of the Control Act**

Control acts are attempts to get another person to act, often with strong social purposes as well. In terms of the action component, we can conceive of a variety of goals in terms of changes in location of goods, activities, and states, varying beneficiaries, and different levels of the conscious attention of the speaker. In common sense language, these variations are coded into terms such as order, offer, prohibition, and so on. For instance, an offer has the addressee as beneficiary.

The point of this topic is to determine what the speaker wants to accomplish through each control act. The first judgment is to decide whether the desired outcome of the act will primarily involve action by the hearer or by the speaker. If the action is by the speaker, then we are talking about offers, intentions, and soliciting permission. If the expected action is by the hearer, we are talking about directives, prohibitions, claims, and permission-giving. The analysis of purpose often can go deeper. We are looking at purpose just behind the surface forms, not in terms of deeper unconscious goals. Try to decide what the goal is for the speaker. This will help you see whether the speaker’s focus is on the goal, on the means, or on what is annoying, that is, on what is to be changed.

**Field 26: Type of Act**

D - Directives. An action is required or requested of the hearer, involving goods or services. Don’t be distracted by the surface form which is coded in Topic 10. The critical issue is WHO IS GOING TO ACT. If it is the addressee, it is a directive.

Can I have some of that?
Is there any of that cake left?
Your brother wants some too.

(cont.)
O - Offers, invitations, or promise of goods or action. We are coding these as control acts because they are initiated by the speaker and affect the hearer. Some benefit to the donor will accrue in almost every case, but to be an offer the speaker must believe that the recipient must also benefit. Of course many speakers will use offers to gain their own ends in a respectable way. One way to determine an offer is to judge how likely a refusal is on the part of the hearer. If the hearer is unlikely to refuse, or if a refusal is due to factors unrelated to the desirability of the offer then a felicitous offer has occurred.

Would you like some of this cake? No, I'm on a diet. Can I help you do that? (where help would benefit addressee) I want to do that for you.

L.-Joint action. A combination of directive and prohibition (J) or directive and intention in which both speaker and addressee are going to equally act (L). Do not use unless you find yourself torn between these two categories.

Get your own fork! (J) Let's watch television. (L)

3.4. Coding and Data Entry

Once a researcher has laid out conventions and methods for doing coding in an easy-to-follow manual, the actual business of coding can begin. This process can be a rather time-consuming task, requiring, not one, but several passes through individual texts. However, there are a few things that coders can do to simplify and speed up the process.

The first step, of course, is for them to familiarize themselves completely with the coding manual. We have found it extremely useful for novice coders to work alongside the researcher on a practice transcript. This experience affords the newcomers the opportunity not only to discuss how to make specific coding decisions but also to check their coding judgments before they move on to code their own texts.

After learning the manual, the second step is for coders to go through their assigned transcripts and to identify all codable cases. In some instances, this can be done from the transcript alone. Frequently, however, coders must do this from the original recorded data. In the case of the CEC, for example, we realized that control acts had to be initially identified from a first pass through the video recording; this could not be done from the transcript alone because outside observers often need to see how actors behave in order to infer a request or prohibition, especially if these moves are not explicit.

The third and final step is to work through the topics for each case, assigning a code to each field, and entering it into a typed or computer record for that coded case. The transcript alone may be sufficient for coding some topics, whereas others may require referring to the audio- or videotape. In using the CEC, for instance, Speaker Identity could be coded from the transcript alone, as could fields relying solely on the surface structure of an utterance, such as Basic Form or Adjunct. In contrast, topics requiring extensive knowledge of context, such as Speech Event, Purpose, and Compliance, had to be coded from audio or video tape. Failure to do this could cause a coder to inaccurately identify the purpose of an act, which requires knowledge of what speakers and addressees do, or to overlook role-play, which requires knowledge of a speaker's manner and voice.

The coding process can be greatly speeded up if coders make a single pass through the tapes to first upgrade their transcripts to include those features that will be coded later on. In the CEC, in fact, we identified 10 topics for which contextual information could be added to a transcript to facilitate coding. Coders were instructed to make a special pass through our videotaped data with an eye toward incorporating features such as tone of voice, addressee gestures, and behavioral compliance. In the process, of course, the richness of the transcript also increased.

A final consideration here is how to minimize "slips of the pen" or mistyping. When data are entered into a computer, it is useful to have a data-entry program that prompts the user for individual fields and checks their entered responses against a set of allowable values for that field.

Our data entry program is called "codeit" (Lampert, 1985). It provides the user with a visual display of the coding record, in which rows represent the individual cases and columns, the topics and fields. When the cursor is moved across the terminal screen from one field to the next, a header appears at the top of the screen to indicate which topic and field the coder has entered, and a line appears at the bottom to indicate all the possible codes that can be entered into that field. If an incorrect code is entered, the editor sends a beep, and an error message appears at the bottom of the screen, instructing the coder to use only the codes indicated. The coder cannot move into the next field unless an acceptable code is entered or the field is left blank.

This code editor, written for a UNIX environment, also allows the user to have selective access to a subset of topics, so that a coder may enter just the codes for a single topic or field in one pass through the data. This can be more efficient for some topics as well as minimize the influence of coding from other fields.
4. EVALUATION

Once coded texts have been produced, the question naturally arises as to whether the codes adequately capture the distinctions they were designed to make. If the guidelines outlined above have been followed, we would expect that we could address the theoretical and empirical questions for which the system was originally designed. There are, however, a few potential problems.

If coders cannot agree on how to classify specific cases, or if it turns out that the codes reflect factors other than the categories they were intended to capture, then the validity of the coding system as well as any coded information resulting from it would certainly be compromised. In this case, it would be difficult, if not impossible, to do any meaningful analyses with the data.

Assuming that codes are reliable and valid, however, there still remain problems regarding how to organize the coding to investigate relationships between variables. Again, improper compiling of the data for analysis would severely compromise the interpretation of any results.

In this section, we address issues that pertain to the reliability of a coding system—that is, the extent to which coders agree on how to characterize individual cases—and in the next we focus on problems and solutions for organizing data for analysis.

4.1. Reliability

One of the most fundamental coding problems is whether or not individuals could code the same cases with one of them assigning codes at random yet, strictly by chance alone, still end up with a reasonably high proportion of agreement.

To illustrate this problem, consider a topic with only two coding options (yes or no) and the following matrix representing the decisions of two coders (A and B) for 100 cases. The columns represent the decisions for coder A, the rows represent the decisions for B, and the cells along the downward sloping diagonal indicate the number of times they agree.

<table>
<thead>
<tr>
<th></th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>75</td>
<td>25</td>
</tr>
<tr>
<td>No</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

This matrix suggests 75% agreement for all 100 cases. However, such an outcome could result with only A making true distinctions across the 100 cases and B indiscriminately coding everything as yes without even considering a single case. When we get patterns of agreement like this one, it becomes difficult to say with any certainty whether agreement is in fact due to carefully weighed judgments or is simply a chance outcome due to a bias for a particular coding category on the part of one or both coders, irrespective of the cases considered.

For any set of cases, we can expect a certain amount of chance agreement. The question, however, is how much is due to chance and how much actually reflects carefully made discriminations.

One descriptive statistic that is used to answer this question is Cohen's Kappa (Cohen, 1960). Kappa or $\kappa$ represents the proportion of observed agreements that is not attributable to chance. This value can be calculated by (a) determining for each coding category the number of agreements that would be expected strictly by chance if one or both coders were assigning codes indiscriminately, (b) subtracting the expected number of chance agreements overall from the actual number of observed agreements, and (c) dividing this difference by the total number of cases minus the number of expected agreements due to chance.

The formula for $\kappa$ is as follows with $O$ and $E$, respectively, representing the number of observed and expected agreements for each category, and $T$ equal to the total number of judged cases.

$$\kappa = \frac{\Sigma O - \Sigma E}{T - \Sigma E}$$

Based on this formula, $\kappa$ would equal 0 for the data in the previous matrix. This suggests that although there appears to be 75% agreement, all of this
agreement is what we would expect from chance alone, based on the distribution of judged cases overall.\footnote{If coders assign categories indiscriminately, then their coding judgments should be statistically independent of one another. Using the probability rules governing independent events, we can calculate the expected number of chance agreements for any category as \( \frac{(A\cap\beta)T}{\beta T} \), where \( A \) and \( \beta \), respectively, are the number of times a coder \( A \) and a coder \( \beta \) use a given category, and \( T \) equals the total number of judged cases. For the yes-no matrix, the expected number of chance agreements overall is

\[
\sum E = \frac{\sum (A\cap\beta)T}{T} = \frac{(75)(100)}{100} + \frac{(25)(0)}{100} = 75
\]

and Kappa equals

\[
\kappa = \frac{\sum O - \sum E}{T - \sum E} = \frac{75 - 75}{100 - 75} = 0
\]

Alternative methods for calculating \( \kappa \) can be found in books on nonparametric statistics \( \text{e.g.}, \) Fleiss, 1981\) as well as Cohen's original article \( \text{Cohen, 1960}. \)

In our own reliability work, we have found values of \( \kappa \) to vary quite dramatically from topic to topic as well as from one coding team to the next. As part of our coding process, we typically have two coders go through a text together to decide jointly which utterances constitute control moves. Once they have done this, however, we then have them code the identified moves separately, and afterward, we compare their work.

We have found that topics requiring relatively objective decision making, such as speaker information or sentential form, tend to produce relatively high levels of coder agreement. For example, we have had proportion agreement as high as \( .97 (\kappa = .95) \) for the topic field General Sentence Type, where coders must decide whether an utterance should be classified as a declarative, interrogative, imperative, or ellipsis. Agreement on this field generally ranges from \( .70 \) to \( .90 \) with \( \kappa \)'s from \( .60 \) to \( .80 \). On the other hand, fields that rely on a larger extent on a coder's intuitions about language tend to reveal lower levels of reliability. For example, we have never found percent agreement for Type of Act to be higher than \( .78 (\kappa = .66) \) with agreement here ranging roughly from \( .60 \) to \( .70 \) with \( \kappa \)'s from \( .50 \) to \( .60 \).

These differences across fields of varying degrees of subjectivity are not surprising. They do, however, emphasize the importance of doing periodic reliability checks to determine where problems lie in the coding system. These checks have helped us in the past to discover coding difficulties with such topics as Purpose, Timing, and Cost, and have led us to rewrite definitions, realign categories, and introduce flowcharts.

In addition to differences across topics, reliability checks have also allowed us to spot differences across pairs of coders. Some pairs, for instance, consistently reveal a high level of agreement from topic to topic, whereas others are consistently low.

In general, these differences could be taken to suggest varying degrees of familiarity with the coding system involved and to indicate the importance of providing similar instructions and training experiences to all coders. In the case of the CEC, however, we believe that these differences more likely reflect a tendency among coders from different backgrounds not to view and interpret language in exactly the same way, irrespective of the amount of training they receive.

In our coding project, in fact, individuals who are assigned to the same text are often trained together, receiving the same instructions and the same practice exercises. However, these individuals often do vary in terms of their familiarity with and similarity to one another, and we have observed anecdotally that coders who are friends and who are from similar backgrounds tend to show a higher rate of agreement than coders who begin as strangers and come from different backgrounds.

This observation is not so surprising given the well-established finding that speakers who are in maximum communication as peers tend to talk alike and to share vocabulary and points of view. Because coders form a social group, the sharing of perspective and the frequency of communication between them should influence the reliability or agreement in their use of the coding vocabulary, and close friends, having had a longer association than most, would be expected to show more agreement in their coding.\footnote{The second author had a vivid experience of this type. The content coding for her dissertation had such a high reliability the faculty committee was suspicious. The reliability coder was her roommate—presumably in better rapport regarding goals and perspectives than hired coders.}

In addition, similarity of social background and orientation could also be expected to increase coder agreement independent of friendship, because of conceptual similarity. If reliability measures are intended to predict probable agreement for a new set of codes, reliability should be tested with a sample of unrelated strangers whose only training is reading the code itself. This would be a very stringent test. In fact, current views of coding reliability consider high reliability to be solely a reflection of the code, rather than of both the code and the specific group of coders used for the reliability assessment.

Given all this, researchers aiming at high reliability measures might be well advised either (a) to hire coders who are quite similar to one another and have an understanding of language similar to the researchers themselves or (b) to get coders from diverse backgrounds but train them to think about language in the same way that the researchers do. We offer, however, what we believe to be a more productive solution in the long run: namely, to hire coders from diverse
4.2. Disagreements and Revision

The hope with respect to reliability, of course, is that after fine tuning a coding system, the level of coder agreement should be or be at least close to 1.00. Perfect coder agreement is so rare, however, that very high agreement arouses suspicion. Reliabilities are typically less than 1.00, and this leads us to an important question: When you do not have perfect agreement, whose codes do you include in the data base?

We have employed two solutions to resolve this dilemma. The first is one of discussion and negotiation. After coders have finished their work, all members of the coding project come together to discuss the difficult cases. The group as a whole goes over and settles on the best way to characterize these cases, and the consensus of the group is then used to decide how to identify these cases in the data base. This method is not only useful for resolving coder disagreements but also for providing another forum for identifying problems for future revisions of the coding system. By identifying the types of coding disagreements that keep recurring, we can identify which coding alternatives are the most difficult to differentiate.

When there are many disagreements, however, the group method becomes impractical, and in these instances, we employ a second, more expedient method in which we assign a third coder to the difficult cases. This third coder goes through only the difficult cases and makes a third set of coding decisions. The codes chosen by two of the three coders (the recoder and the original two) are then selected for inclusion in the data base. If it turns out that all three coders disagree on how to code a particular case, then this case is turned over to the project group, and the first method is used to resolve the disagreement.

In all such cases, it is important to keep a record of what has happened and to use the experience to improve the examples in the code and to reorganize categories.

5. APPLICATIONS

Once a data base of coded information has been assembled, the final question is how to put it to use. Assuming that the codes have been laid out in some systematic fashion, there are two possibilities here. The first is to use the codes to search for all cases of a particular type for a more qualitative examination; the second is to use them to perform statistical analyses on the relationships between coding dimensions.

5.1. Codes and Qualitative Research

Regarding searches, researchers often wish to retrieve similarly coded lines of text so that they can be systematically compared with one another. Searches with this purpose in mind can be easily accomplished by providing an index with each line of coding that identifies the line of text that corresponds to it. In the CEC, for example, we lay out our codes in single-line records with an index that identifies the location of the original line of text at the head of each record. When we wish to search for utterances of a particular type, for instance, prohibitions, we simply skim through our data base and locate all the code lines with P in the topic field Type of Act and then use the record index to find the corresponding prohibitions in the original transcripts.

In addition to indexing, we have also done two other things with our computerized data base. First, we have developed our own text-scanning software, labeled “textsca” (Lampert, 1986). This software allows us to specify any pattern of text and returns all lines containing these patterns. What makes this search program unique compared to others is that it allows the user to specify what topic fields to search and which code letters to match in those fields, provided that the fields are in the same fixed location from case to case. Because multiple fields can be scanned together, the program can search for and return any pattern of coded information.

To simplify searches, we have also found it useful to interleave our codes and transcripts to create computer files that contain lines of text followed by the corresponding records of coded information. The original code and transcript files are left intact for other purposes. The new files are then used for searches with textsca, which has the capability to match any line and to return up to 12 of the lines before and after the matched one. This feature allows us to recover the context in which a match occurs.

5.2. Codes and Quantitative Research

In addition to aiding searches, codes can also be used to perform statistical analyses. Currently, there are a number of statistical packages available for calculating various descriptive statistics (i.e., means, standard deviations, correlation coefficients, etc.) and performing inferential tests. Along with the Statistical Package for the Social Sciences mentioned earlier, these include the Statistical Analysis System (SAS) and the Biomedical Computer Programs-P series (BMDP).
Many of these packages require that the values for each variable appear in the same location from case to case. If a coding system assigns codes to fields with fixed locations, this does not pose a problem: Most statistical packages can read an entire data base provided that every field's location and length are included as input for each program. In short, if a data base of codes is highly structured to begin with, it would appear that very little has to be done to prepare it for computer analysis. However, this is not necessarily the case.

Whereas an original data base may not have to be realigned for computer analysis, some reorganization is often necessary to create variables that can be meaningfully studied. Reorganization is especially needed if the researcher wishes to compare the language of different groups of people. In such a case, the researcher must have records that reflect not only which categories were used from case to case but also how many times any given individual employed a particular category.

This is a serious consideration, for once a data base has been compiled, the temptation often is to simply correlate one topic field with another. This procedure would be fine if all the data came from a single individual and all the researcher wished to do were to show relationships between variables (i.e., form and purpose) for just one person. However, when we wish to make statements about groups of people, we have to take into account individual differences, as the unusual behavior of just one person can distort the picture of the group as a whole. This problem is best illustrated by example.

Suppose, for instance, that we were interested in whether 4-year-olds used more hints to make requests than 3-year-olds (hims are picked up in the CEC under the topic Explicitness) and that we had four children at each age level with each level providing 200 coded cases. To address our research question, we could tally up and compare the total number of hints for the two groups. If we did this, we might then discover that the 3-year-olds and the 4-year-olds produced a total of 28 hints each. Such an outcome might suggest no relationship between age and hinting. However, if we inspected the data a little closer we might have discovered that although each group produced the same number of hints, each 4-year-old produced 7 hints whereas a single 3-year-old produced all 28 for the younger age group. This outcome suggests a totally different picture. In fact, if we went by the number of children who actually provided hints, we might conclude that there was an age trend (inferential tests aside).

In short, data that do not reflect individual performance can lead a researcher to conclude no group differences when differences do exist and, conversely, to claim differences when there are none at all. This problem of treating individual observations as if they each were independent of one another and came from a different person is not new to the social sciences. In fact, textbooks in behavioral statistics often admonish students not to do this and refer to this problem as the case of the inflated N (see Runyon & Haber, 1991).

A language researcher can avoid having an inflated N by simply providing separate scores for speakers across coded transcripts. In fact, it is useful to have individual speaker scores or totals for each topic area that will be analyzed. However, even individual totals are often not adequate for analysis, for these too can be artificially inflated (or deflated) in natural language research as a function of the length of observation or involvement. For example, two children may be equally likely to use indirect requests; however, because one is observed for 1 hour and the other for 2 hours, the latter may appear more likely to hint when the total number of indirect requests are finally tallied. This particular problem can, in fact, be exacerbated when longer observations occur and longer transcripts are produced for all individuals in one comparison group as opposed to those in another.

The solution to this problem is often to weight the individual totals in some fashion; that is, to create a score that reflects the number of instances for a given behavior per unit of time or some other common denominator that, in language research, could be number of utterances or turns. For instance, if we were trying to compare the frequency of hinting for two children in similar situations, observed for 1 and 2 hours, respectively, we could divide the number of hints each produced by the amount of time each was observed to obtain two scores reflecting the number of hints per hour. In this way, the length of observation can be controlled.

If each code line in a data base provides fields to identify the speaker and the amount of time the speaker was observed within a particular transcript, or a particular activity within a transcript, these fields can be used to create weighted totals for individual speakers. Most statistical packages, in fact, can be programmed to create these totals without the user having to manually realign the original data.

Once individual totals have been assigned, these can then be analyzed according to age, sex, or any other subject or contextual variable. Group means and standard deviations can be calculated, and parametric and nonparametric tests (e.g., analysis of variance, Kruskal-Wallis test, Friedman test, etc.) can be performed. For further information on how to calculate statistics and carry out the various inferential tests, we suggest Hatch (1982), Hays (1988), Marascuilo and McSweeney (1977), Marascuilo and Serlin (1988) and Woods, Fletcher, and Hughes (1986).

---

*Providing each case with an index for the amount of time that the speaker was engaged in a current activity has the advantage of allowing the researcher the opportunity to study weighted means for any activity type. If only the length of the transcript is recorded, this would not be possible, as total time observed is likely to be longer than the time in any given activity, leading to an inaccurate weighting.*
Needless to say, statistical analyses are important for uncovering relationships among variables, leading to lawful generalizations. However, they also play an important role in the evolution of a coding system. As we have stated throughout, most coding systems are guided by either an explicit or, at least, implicit theory about language. By analyzing coded data, researchers have an opportunity to evaluate whether there is any empirical support for their original hypotheses. Sometimes these will be borne out; sometimes they will not. Lack of support, however, may suggest either that the original theory needs revision or that the coding system needs to be reorganized to be more sensitive.

In the case of the CEC, for example, our original code did not include the topic of cost. When we initially analyzed the data in terms of the relation of politeness of request to compliance, we found that there was a very strong relationship: the more polite the request, the less the likelihood of compliance. We realized that what we had overlooked was that both politeness and noncompliance with requests increase when what is requested is difficult, expensive, time-consuming, or valuable to the addressee. As a result, we developed the variable of cost to control for this factor and found that when cost was low to moderate, politeness was indeed unsuccessful in gaining compliance (Ervin-Tripp, Guo, & Lampert, 1990; Ervin-Tripp, O'Connor, & Rosenberg, 1984), and in fact increased refusals by most listeners, especially adults. Only high-cost requests to peers and older children were more effective if polite.

We expected that requests directed toward a superior would reveal greater deference, and for the most part, this is what we found. While doing pilot work with the cost code, however, we were further surprised in discovering that, counter to our initial hypotheses, young children's directives to mothers (but not fathers) were quite direct and showed little deference. This led us to rethink the definition of cost, at least as it applied to caregivers. We eventually modified the code so that requests for things that were within the addressee's cultural role to provide were coded as having a neutral cost.

What we have hoped to illustrate by this example is that once again the code-building process does not necessarily have a specific end, but rather can be ongoing, involving revisions even during and after a phase of statistical analysis.

Coding and statistical analysis of coded material are not, of course, the only ways one can analyze verbal discourse. This type of analysis can be done either before or after qualitative studies that provide a more vivid account of specific cases and reveal more clearly the tactical choices that speakers make. Coding and statistical analysis done after qualitative studies can then provide reliable tests or confirmation of patterns found in the qualitative studies, as well as suggest additional hypotheses for further testing.

6. CONCLUSION

In this chapter, we have attempted to outline the evolution of a structured coding system for the study of natural language by taking the reader through four separate phases of the coding process: construction, implementation, evaluation, and application. We have presented some of the basic problems that researchers encounter when developing a new coding system and have offered some solutions to these problems based on our own experiences with the Control Exchange Code. We have also attempted to illustrate how many of the problems and solutions encountered in instituting and learning a code have much in common with the learning of a second language. Our hope in doing this is that we have provided not only insight into the coding process but also have offered some ideas on how to organize future coding projects.

REFERENCES


Coding Child Language Data for Crosslinguistic Analysis

Dan I. Siobin
University of California at Berkeley

1. INTRODUCTION

In carrying out crosslinguistic research, the investigator is faced with all of the issues raised in this volume, multiplied by two or more. Data in each of the languages involved in a crosslinguistic project must be transcribed, coded, and analyzed. But, in addition, the descriptive categories used in these tasks must be applicable in comparable fashion across languages. Comparability applies to both form and function. Put most simply, a researcher needs a uniform coding system that facilitates the comparison of language-specific forms with respect to common functions.

We can schematize the task as involving three broad types of information:

1. utterances and accompanying nonverbal acts, with their context (as discussed in Part I of this volume)
2. coding categories for linguistic forms (phonology, morphosyntax, prosody)
3. coding categories for functions (semantic, pragmatic)

Transcription practices in crosslinguistic studies are the same as for single-language studies, except for the possible addition of an English gloss of the utterance. In principle, the analytic categories for forms and functions should be drawn from “universal” theories, applicable to any and all studies. In practice, though, special care must be paid to ensure comparability across languages within the study at hand. Ideally, of course, all single-language studies would be comparably coded as well, allowing for future crosslinguistic