Codebook Development for Team-Based Qualitative Analysis

Kathleen M. MacQueen
Centers for Disease Control and Prevention (CDC), Atlanta
Eleanor McLellan
TRW, Inc., Atlanta
Kelly Kay and Bobby Milstein
CDC, Atlanta

Cultural Anthropology Methods 10(2):31-36 1995

One of the key elements in qualitative data analysis is the systematic coding of text (Strauss and Corbin 1990:57–60; Miles and Huberman 1994:56). Codes are the building blocks for theory or model building and the foundation on which the analyst's arguments rest. Implicitly or explicitly, they embody the assumptions underlying the analysis. Given the context of the interdisciplinary nature of research at the Centers for Disease Control and Prevention (CDC), we have sought to develop explicit guidelines for all aspects of qualitative data analysis, including codebook development.

On the one hand, we must often explain basic methods such as this in clear terms to a wide range of scientists who have little or no experience with qualitative research and who may express a deep skepticism of the validity of our results. On the other, our codebook development strategy must be responsive to the teamwork approach that typifies the projects we undertake at CDC, where coding is generally done by two or more persons who may be located at widely dispersed sites. We generally use multiple coders so that we can assess the reliability and validity of the coded data through intercoder agreement measures (e.g., Carey et al. 1996) and, for some projects, as the only reasonable way to handle the sheer volume of data generated. The standardized structure and dynamic process used in our codebook development strategy reflects these concerns.

This paper describes (1) how a structured codebook provides a stable frame for the dynamic analysis of textual data; (2) how specific codebook features can improve intercoder agreement among multiple researchers; and (3) the value of team-based codebook development and coding.

Origins of the Codebook Format

Our codebook format evolved over the course of several years and a variety of projects. The conceptual origins took shape in 1993 during work on the CDC-funded Prevention of HIV in Women and Infants Project (WIDP) (Cotton et al. 1998), which generated approximately 600 transcribed semistructured interviews. One research question pursued was whether women's narratives about their own heterosexual behavior could help us understand general processes of change in condom use behavior (Milstein et al. 1998). The researchers decided to use the processes of change (POC) constructs from the Transtheoretical Model (Prochaska 1984; DiClemente and Prochaska 1985) as a framework for the text analysis. However, the validity of the POC constructs for condom-use behavior was unknown, and a credible and rigorous text coding strategy was needed to establish their applicability and relevance for this context.

To do this, the analysts had to synthesize all that was known about each POC construct, define what it was, what it was not, and, most importantly, learn how to recognize one in natural language. Several years earlier, O'Connell (1989) had confronted a similar problem while examining POCs in transcripts of psychotherapy sessions. Recognizing that "coding processes of change often requires that the coder infer from the statement and its context what the intention of the speaker was," O'Connell (1989:106) developed a coding manual that included a section for each code titled "Differentiating (blank) from Other Processes." Milstein and colleagues used O'Connell's "differentiation" section in

a modified format in their analysis of condom behavior change narratives. They conceptualized the "differentiation" component as "exclusion criteria," which complemented the standard code definitions (which then became known as "inclusion criteria").

To facilitate on-line coding with the software program Tally (Bowyer 1991; Trotter 1993), components were added for the code mnemonic and a brief definition, as well as illustrative examples. Thus, the final version of the analysis codebook contained five parts: the code mnemonic, a brief definition, a full definition of inclusion criteria, a full definition of exclusion criteria to explain how the code differed from others, and example passages that illustrated how the code concept might appear in natural language. During the code application phase, information in each of these sections was supplemented and clarified (often with citations and detailed descriptions of earlier work), but the basic structure of the codebook guidelines remained stable.

A similar codebook format was used in an analysis of brief open-ended responses to a question about perceived exposure to HIV, which was included in a standardized survey of 1,950 gay and bisexual men enrolled in the CDC Collaborative HIV Seroincidence Study (MacQueen et al. 1996). For this project we needed a codebook that would facilitate coder recognition of discrete events and specific behaviors as well as the ability to distinguish statements of fact (e.g., a partner who was *known* to be HIV-seropositive) from statements of perception (e.g., a partner who was *thought* to be uninfected). More recently, we have used the structured codebook format as a key element in data management and analysis for Project LinCS: Linking Communities and Scientists, a multisite study with over 300 transcribed in-depth interviews (MacQueen and Trotter 1997). Thus far, we have developed seven distinct codebooks to guide and structure our use of the extensive Project LinCS data base. In addition, our codebook format has been incorporated into in-house software applications for qualitative data management and analysis such as CDC EZ-Text (Carey et al. 1998).

Codebook Structure

The codebook structure has evolved to include six basic components: the code, a brief definition, a full definition, guidelines for when to use the code, guidelines for when not to use the code, and examples. Table 1 illustrates how the components were operationalized for a code we developed as part of an analysis of emic representations of "community" (Blanchard et al. 1997).

Though the structure is simple and stable, the process of building the codebook is complex and dynamic. We have found that a relational database management program such as Access®[1] (Microsoft Corporation) is conceptually well suited for both maintaining the structure and supporting the process. In a relational database the basic unit is the *relation*, which is generally conceptualized as a table with rows of "observations" (in our case, the codes) and columns of "attributes" (the definitional parameters for the codes). The rows and columns look like a standard quantitative data set, and the format provides all of the flexibility and automated processing typically associated with quantitative data management. However, unlike many database management systems that are specifically designed for quantitative data, the cells in a table or relation in Access® can contain unlimited amounts of text. This feature makes it possible to work with the codebook as a qualitative database.

There are practical advantages to using a data base management program for codebook development. First, it facilitates a systematic approach by providing a template that prompts for key components. Second, it facilitates the production of multiple versions of the codebook, for example, a simple listing of the codes with brief definitions for quick reference and a full version with all components for detailed reference and comparison of coding guidelines. Third, it is easy to revise the codebook and then print only the modified codes and their definitions (rather than the whole codebook). Fourth, use of a date field makes it easy to generate reports that list coding guidelines that were changed after a particular date. Fifth, the codebook can be output in a wide variety of formats including ASCII text files, spreadsheets, and other database formats. Some of these formats may then be imported directly into qualitative analysis programs such as Tally, NUD*IST (Qualitative Solutions & Research), and ATLAS/ti (Scientific Software Development). Sixth, the fact that the codebook can be detached from other parts of the analytic database means it can easily be modified, copied, and replaced.

Last, but not least, a codebook expressed as a relation or table in a relational database can, in turn, be linked to other codebook tables. This allows the analyst to define directional links between sets of codes and add new dimensions to the relationships among codes. For example, hierarchical relationships can be defined such that a single code in one table is linked to a series of related codes in another table (a "one-to-many" relationship). Hierarchical relationships can be used to create families of codes (Muhr 1994) that can be aggregated, reviewed, and analyzed at increasingly general levels. Thus, the scale of analysis can be easily and systematically modified. Alternatively, a set of codes in one table may have multiple links to those in other tables, forming a "many-to-many" relationship. In this case, the code relationships can be summarized as a network. Both hierarchical and network linkages can be portrayed graphically.

In our approach, the codebook functions as a frame or boundary that the analyst constructs in order to systematically map the informational terrain of the text. The codebook may or may not reflect "natural" boundaries for the terrain, but it always reflects the analyst's implicit or explicit research questions. It forces the analyst to place his or her assumptions and biases in plain view. The codes, in turn, function like coordinates on the map frame; when applied to the text, they link features in the text (e.g., words, sentences, dialog) to the analyst's constructs. The adequacy of answers to research questions can then be assessed in terms of the sensitivity and specificity of the codes, the richness of the text, and the validity and reliability of the links established among them. From this perspective, the central challenge of systematic qualitative analysis lies within the coding process.

The Coding Process

As Bernard (1994:193) points out, a code can be used as an encryption, indexing, or measurement device; we focus on using index codes to tag text for retrieval and measurement codes to assign values to text such as the frequency, amount, or presence/absence of information (Bernard 1994; Seidel and Kelle 1995; Bernard and Ryan In press). In both cases, the code adds information to the text (rather than reducing the text) through a process of interpretation that simultaneously breaks the text down into meaningful chunks or segments. Thus, the coding process must include explicit quidelines for defining the boundaries of the text associated with a particular code. How the text is segmented is influenced by the data collection strategy and the way that strategy structures the resulting text. Data collection strategies generally fall along a continuum from minimally structured (e.g., transcribed interviews) to maximally structured (e.g., brief responses to open-ended survey questions).

With large databases comprised of in-depth transcribed interviews we find it useful to begin by coding text according to the specific research questions used to frame the interview; we label this type of index coding as Stage 1 or *structural coding*. The purpose of this step is to facilitate subsequent analysis by identifying all of the text associated with a particular elicitation or research question. The code inclusion criteria can include linguistic cues that signal the coder to apply structural codes to text that is out of sequence with the original interview structure. This is important for situations where respondents spontaneously return to earlier topics or make a cognitive leap to a topic that the interviewer intended to cover later. With regard to segmenting the text for structural coding, we find it useful to include within each segment the full elicitation from the interviewer and the full response from the participant, including all dialog between the interviewer and participant that flows from the elicitation. This preserves both the flow of the interview and the full context of the discussion on a particular topic.

Structural coding generally results in the identification of large segments of text on broad topics; these segments can then form the basis for an in-depth analysis within or across topics. We have found that code definitions that incorporate substantial references to professional jargon (i.e., etic codes) tend to encourage the use of coders' preconceptions in the analysis, making it difficult to distinguish the text (or "voice") of the respondent from that of the analyst or researcher. In contrast to structural coding, we therefore attempt to use respondents' own terms and semantics to guide the construction of codes and their definitions for in-depth analysis (i.e., emic codes). In addition, the use of linguistic cues to assist coders in correctly applying codes to text tends to reduce the potential for misinterpretation and omission of relevant information.

For example, in one of our projects we created three socioeconomic class status codes (poor, middle class, and rich) to measure the salience of socioeconomic class for the way individuals perceived the structure of their community. This was based on an etic or "objective" viewpoint. However, these codes did not capture the more subjective viewpoint of

the participants, who often described the financial status of others relative to their own, e.g., richer, poorer, "those with more money." The etic codes and definitions also led the coders to make implicit assumptions about education, employment, access to resources, and basic living conditions that were not always supported in the text. We therefore eliminated the etic codes with their implicit assumptions about social conditions and used a single code to capture all references to income levels and socioeconomic class, with additional codes to capture explicit references to homelessness and employment status.

Text segmentation during in-depth analysis is less straightforward than during structural coding. A coded segment could be as simple as a text marker or tag placed over a word or phrase, with the boundaries of the segment free floating; in this case the segment can be continuously redefined during analysis to include only the word, a set number of lines above and below the word, or the full text document. Alternatively, rules for bounding segments can be established a priori, e.g., through use of textual units such as sentences or grammatical guidelines such as requiring the inclusion of subject references.

With brief responses to open-ended questions on standardized surveys there is generally little need to develop structural codes because the data is prestructured by question and participant. Here the goal is to code the text in such a way that the information can be combined meaningfully with a quantitative database. The codes are used primarily to signal the presence or absence of particular pieces of information. The open-ended responses can then be summarized in a 0/1 matrix where the rows are labeled with participant identification numbers and the columns with the codes; the cells are filled with either a "0" (to indicate that the information was not present) or a "1" (to indicate that it was present). For example, Table 2 presents a matrix for a hypothetical group of responses to a question concerning how persons think they may have been exposed to HIV. It would also be possible to attach values to the codes, for example, the MULTPARTS code could be valued to reflect the total number of partners that a person reported (provided that information is available from the data).

In order to facilitate quantitative analysis of open-ended responses it is generally advisable to limit the total number of codes to be used. However, this can easily be done by collapsing code categories together through data transformations performed on the matrix. The final code categories can be built into the coding process through the use of hierarchical codes and code families. Such a strategy is preferable to limiting the content of the codebook a priori only to find that the resulting codes are too crude to permit a meaningful analysis.

There are a several software programs available that support the analysis of open-ended structured survey data. CDC EZ-Text (Carey et al. 1998) is specifically designed for text entry and coding of brief open-ended responses; as noted previously, it includes the same codebook structure outlined in this paper. It also includes matrix building capabilities and an export feature that transforms the coded data into a quantitative database that can be imported into a number of statistical programs for further analysis. TextSmart™ (SPSS Inc.) is a new program that uses an automated content-analysis approach for quantitative analysis of very brief responses (<255 characters). We have not explored the utility of the structured codebook for this approach to coding.

Between the examples discussed above (transcribed interviews and open-ended survey questions) lie a range of other possibilities. For example, rather than record and transcribe interviews verbatim, researchers may take notes and then organize their notes according to specific research topics. Although the elicitation is relatively unstructured, the resulting text is highly structured and presegmented by research topic. Another possibility would be the collection of open-ended responses on a single question across multiple waves of structured interviews with the same individuals. The responses from each wave could be compiled into a single textual response for each individual that permits segmentation by recurrent themes as well as by interview wave. The analysis of other types of text such as field notes, diaries, and newsletters each present their own challenges and possibilities. The issues of data collection and text segmentation as elements in the coding process are complicated and deserve further systematic exploration.

Refining the Codebook

Before coding an entire data set, we systematically evaluate the utility of the codes and the coders' ability to apply the codes in a consistent manner. The steps in this process begin with the development of an initial code list derived from

etic concepts, emic themes, or both (Figure 1). Usually, one or two team members take a lead role in developing the list, which is then reviewed by the full research team (Figure 1). Once there is agreement on the scope and level of detail for the items in the list, the team leaders begin the process of codebook development. Definitions are proposed and reviewed by the team, with an emphasis on achieving clarity and explicit guidance for code application.

When the team is comfortable with the code definitions, two or more coders are then given the task of independently coding the same sample of text. The results of their coding are then compared for consistency of text segmentation and code application. If the results are acceptable and consistent, the coding continues with periodic checks for continued intercoder agreement. If the results are unacceptable and inconsistent, the inconsistencies are reviewed by the coders and team leader(s).

The codebook is reviewed to determine whether the inconsistencies are due to coder error, e.g., misunderstanding of terminology or guidelines. We view these as training errors and they are generally handled by the coders and team leader(s). Other inconsistencies are due to problems with the code definitions, e.g., overlapping or ambiguous inclusion criteria that make it difficult to distinguish between two codes. These types of problems are generally discussed by the whole team, as they have implications for the interpretation of the text. Once the problems are identified and the codebook clarified, all previously coded text is reviewed and, if necessary, recoded so that it is consistent with the revised definitions. Intercoder agreement is again checked, to ensure that the new guidelines have resolved the problem. This iterative coding process continues until all text has been satisfactorily coded.

There are a variety of ways that intercoder agreement can be assessed. For transcribed interviews, we generally prefer a detailed segment-by-segment review that includes assessments of consistency in defining the beginning and end of segments as well as the application of codes within segments. All inconsistencies are noted, discussed, and resolved. When only a subsample of the text is coded by multiple coders, this strategy may not capture all coding errors. In particular, it is prone toward the loss of pertinent text, that is, text that is not captured by any code. Erroneously captured text, in contrast, is more likely to be spotted during the course of either coding checks or subsequent analysis.

As noted previously, we also use kappa to statistically assess the level of intercoder agreement. This approach is useful for identifying coder error, as long as the error is not due to systematic mistraining of all coders. It is also helpful for identifying poorly defined codes, which tend to be applied erratically.

Some Practical Suggestions

We have learned eight practical lessons from our experience with team-based codebook development and application.

1. Assign primary responsibility to one person for creating, updating, and revising a given codebook. This will ensure that the codebook is consistent, and it will minimize ambiguities due to differences in vocabulary and writing styles. Require all members of the coding team to clear all coding questions and clarifications with the codebook editor. Make certain that the codebook editor has the basic competence for the task.

2. Schedule regular meetings where the coding team reviews each code and definition in the codebook. It is easy for a coder to develop a set of implicit rules without realizing that the codebook no longer reflects his or her actual coding process; in addition, this evolving process may or may not be shared by other members of the coding team. Without regular review sessions, the codebook can quickly become obsolete and useless as an analytic tool.

3. Establish a coding process that consciously seeks to enhance intercoder agreement. One way to do this is to create a codebook that someone can learn to use within a few hours. For the most part, coders can reasonably handle 30–40 codes at one time. If the codebook contains more than 40 codes, the coding process needs to be done in stages. Separate the codebook into three or four related components or domains and have the coders apply one set of codes to the entire data set before moving onto the next set.

4. Develop a written plan for segmenting text at the same time the codebook is being developed. Segmenting complex

text into smaller natural units, such as a line or a sentence, may be helpful. If you are segmenting less than a sentence, it may be better to utilize a word-based content analysis strategy than a codebook strategy. Of course, the word-based strategy can evolve into codebook development.

5. Establish intercoder agreement measures early in the process of codebook development. Decide a priori how much variability you are willing to permit with regard to both segmentation and code application, and how you will assess that variability. Make sure the coding team understands and agrees with the criteria, and that coding will not be considered final until those criteria are met.

6. When defining codes, do not assume that anything is obvious; always state specifically what the code should and should not capture. This includes defining common abbreviations and "short hand" references that may occur in the text. Include all such information in the full definition of the code and explanations for its use. Things that seem obvious to coders at a certain place and time can be totally obscure in another context, even to the same coders.

7. Don't clutter the codebook with deadwood: throw out codes that don't work, and rework definitions for codes that are problematic. Some codes may capture the specificity of a single response rather than general patterns and themes. It is best to eliminate these types of codes from the codebook or expand their usage. A single generic code (e.g., UNIQUE) can be designed to capture all unique responses on a given topic. The codebook should be a distillation, not an historical document. If maintaining a history of the codebook development process is relevant, then develop a separate strategy for that purpose.

8. Finally, accept the fact that text will need to be recoded as the codebook is refined. Recoding should not be viewed as a step back; it is always indicative of forward movement in the analysis.

Conclusion

The interdisciplinary team-based approach to qualitative analysis at CDC has provided us with both the incentive and the resources to develop explicit guidelines for and documentation of our methods. For coding, this has led to the generation of a basic structure for organizing codebooks that is also flexible enough to meet the needs of a variety of coding situations. When combined with an informed discussion of the coding process, the use of a flexible yet standard structure facilitates coder training. It also enhances the analyst's ability to transfer skills from one project to another, regardless of variability in the particular software tools used.

Acknowledgments

We thank Gery Ryan, Jim Carey, and Linda Koenig for their helpful suggestions and comments on earlier drafts of this paper. Many thanks to Bob Trotter and the Project LinCS collaborators who patiently field tested and critiqued the methods described here.

Notes

1. Use of trade names is for identification only and does not imply endorsement by the Public Health Service or by the U.S. Department of Health and Human Services.

References

Bernard, H. R. 1994. *Research Methods in Anthropology: Qualitative and Quantitative Approaches,* 2nd Ed. Walnut Creek, CA: AltaMira Press.

Bernard, H.R. and G. Ryan. In press. Qualitative and Quantitative Methods of Text Analysis. In *Handbook of Research Methods in Cultural Anthropology*, H. R. Bernard, editor. Walnut Creek, CA: AltaMira Press.

Blanchard, L. 1997. How Do You Define Community? Perspectives of Community Members. Paper presented at the American Anthropological Association Annual Meeting, November 19–23, Washington, DC.

Bowyer, J. W. 1991. *Tally: A Text Analysis Tool for the Liberal Arts.* Dubuque: William C. Brown Publishers.

Carey, J. W., M. Morgan, and M. J. Oxtoby 1996. Intercoder Agreement in Analysis of Responses to Open-Ended Interview Questions: Examples from Tuberculosis Research. *CAM 8*(3):1–5.

Carey, J. W., P. H. Wenzel, C. Reilly, J. Sheridan, and J. M. Steinberg. 1998. CDC EZ-Text: Software for Management and Analysis of Semistructured Qualitative Data Sets. *CAM 10*(1):14-20.

Cotton, D. A., R. J. Cabral, S. Semaan, and A. Gielen. 1998. Prevention of HIV in Women and Infants Projects. Application of the Transtheoretical Model for HIV Prevention in a Facility-Based and a Community-Level Behavioral Intervention Research Study. Manuscript submitted for publication.

DiClemente, C. C. and J.O. Prochaska. 1985. Processes and Stages of Self-Change: Coping and Competence in Smoking Behavior Change. In *Coping and Substance Abuse,* S. Schiffman and T. A. Wills, eds. Pp. 319–343. New York: Academic Press.

MacQueen, K. M., K. L. Kay, B. N. Bartholow, et al. 1996. The Relationship Between Perceived Exposure to HIV and Actual Risk in a Cohort of Gay and Bisexual Men. Poster presentation at the XI International Conference on AIDS, July 7–12, Vancouver, BC, Canada.

MacQueen K. M. and R. T. Trotter. 1997. Project LinCS: A Multisite Ethnographic Design for Public Health Research. Paper presented at the American Anthropological Association Annual Meeting, November 19–23, Washington DC.

Miles, M. B., and A. M. Huberman 1994. *Qualitative Data Analysis*, 2nd ed. Thousand Oaks, CA: Sage Publications.

Milstein, B., T. Lockaby, L. Fogarty, A. Cohen, and D. Cotton. In press. Women's Processes of Behavior Change for Condom Use. *Journal of Health Psychology.*

Muhr, T. 1994. *ATLAS/ti—Computer Aided Text Interpretation & Theory Building, Release 1.1E. User's Manual*, 2nd ed. Berlin: Thomas Muhr.

O'Connell, D. 1989. Development of a Change Process Coding System. Ph.D. diss., University of Rhode Island, Kingston, RI.

Prochaska, J. O. 1984. *Systems of Psychotherapy: A Transtheoretical Analysis*, 2nd ed. Homewood, IL: Dorsey Press.

Seidel, J and U. Kelle. 1995. Different Functions of Coding in the Analysis of Textual Data. In *Computer-Aided Qualitative Data Analysis: Theory, Methods, and Practice.* Udo Kelle, ed. Pp. 52–61. Thousand Oaks, CA: Sage Publications.

Strauss, A. and J. Corbin. 1990. *Basics of Qualitative Research: Grounded Theory Procedures and Techniques.* Newbury Park, CA: Sage Publications.

Trotter, R.T. 1993. Review of TALLY 3.0. *CAM* 5(2):10-12.