AUTHOR INFORMATION

Megan Skillman
55 East Monroe Street
30th Floor
Chicago, IL 60603
(312) 357-3875
Skillman-Megan@norc.org

Rachel Singer
4350 East-West Highway
8th Floor
Bethesda, MD 20814
(301) 634-9321
Singer-Rachel@norc.org

Christina Rotondo
4350 East-West Highway
8th Floor
Bethesda, MD 20814
(301) 634-9312
Rotondo-Christina@norc.org

Sarah Ruiz
National Institute on Disability, Independent Living, and Rehabilitation Research
Administration for Community Living
U.S. Department of Health and Human Services
330 C Street SW
Washington, DC 20201
(202) 795-7782
Sarah.Ruiz@acl.hhs.gov

Adil Moiduddin
4350 East-West Highway
8th Floor
Bethesda, MD 20814
(301) 634-9419
Moiduddin-Adil@norc.org
# Table of Contents

Introduction ................................................................................................................................. 1

Choosing an analytic approach ................................................................................................. 3
  I. Inductive coding .................................................................................................................. 3
  II. Deductive coding ............................................................................................................. 4
  III. Combined approaches .................................................................................................... 4

Codebook Development, Coder Training, and Refinement of Themes ................................. 5
  Training a large team ......................................................................................................... 7
  Cross-coding and code complexity .................................................................................... 8
  Quality assurance .............................................................................................................. 9

Adapting to Rapid-Cycle Analysis and Reporting ................................................................. 10
  Identifying research priorities .......................................................................................... 10
  Data preparation ............................................................................................................. 10
  Use of computer-assisted qualitative data analysis software ......................................... 10
  Staging coding ................................................................................................................ 11

Discussion ................................................................................................................................. 11

References ................................................................................................................................. 13

## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Summary of Qualitative Data Collection</td>
<td>2</td>
</tr>
</tbody>
</table>

## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Code Families and Codes</td>
<td>6</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Simultaneous Coder Training and Code Refinement</td>
<td>6</td>
</tr>
</tbody>
</table>
Introduction

With the proliferation of innovations and pilot programs created under Affordable Care Act initiatives, there is increasing pressure to understand interventions and their success as early as possible. Rapid-cycle evaluation provides an “approach to assessing the effectiveness of interventions more rapidly and to a philosophy of providing ongoing feedback to participating providers to support continuous quality improvement” (Shrank 2013). Formation of the Rapid Cycle Evaluation Group at the Centers for Medicare & Medicaid Services (CMS) Center for Medicare & Medicaid Innovation (CMMI) shows the future potential for this style of evaluation to inform policy and program replication.

Previous discussion of rapid-cycle evaluation has occurred in the context of electronic health data analytics (Schneeweiss et al. 2015), creating efficient data collection (Hamilton 2013), and general justification for the approach (Parry et al. 2013), but there is little guidance in the literature on qualitative analysis in a rapid-cycle evaluation context. The purpose of this paper is to describe some of the practical strategies and lessons learned from rapid-cycle qualitative analysis conducted as part of a multisite evaluation under the CMMI Health Care Innovation Award (HCIA) initiative.

Evaluation of the Health Care Innovation Awards (HCIA). The Health Care Innovation Award (HCIA) program supports the testing of new care-delivery approaches, including those that leverage technical applications, workforce training, deployment of new delivery models, and ongoing improvement informed by rapid-cycle feedback. In 2012, CMMI awarded $900 million to 108 programs across the country to test new payment and service delivery models to improve the quality of care and lower costs for Medicare, Medicaid, and/or CHIP enrollees (CMS 2016). NORC at the University of Chicago conducted a mixed-method rapid-cycle evaluation of a subset of 18 programs targeting patient populations with specific diseases or diagnostic profiles. The evaluation sought to understand (1) the HCIA interventions (e.g., implementation, evolution); (2) the impact of interventions (e.g., cost, health care utilization, quality of care); (3) the contexts that contributed to successes and facilitators; and (4) how workforce initiatives contributed to program effectiveness.¹

The quantitative evaluation assessed the relationship between awardee programs and measures of health, quality of care, and health care costs and utilization. Data sources included: claims data for Medicare or Medicaid beneficiaries, depending on the primary population the awardee serves; and awardee-collected

data, which includes administrative program data, electronic health record data, clinical measures, surveys, and other participant-reported outcomes.

Qualitative data played an important role by lending insight into how programs worked and led to hypotheses about what quantitative outcomes might look like. Using complementary qualitative coding structures allowed us to explain quantitative findings, such as why health care utilization may or may not be reduced, or describe themes that are difficult to measure with quantitative data, such as confidence managing chronic conditions or perceived improvements in quality of health care among participants.

Qualitative analysis also identified potential drivers behind programs’ successes or lack of success addressing challenges.

From March 2014 to December 2015, we conducted two rounds of site-based data collection that included semi-structured interviews of frontline program staff (e.g., care coordinators, community health workers, physicians), leadership (e.g., supervisors, site managers, principal investigators), and partners (e.g., clinics, community based organizations) totaling 559 individuals (NORC Disease-Specific 2015). Data collection also included 40 focus groups and 134 one-on-one patient and/or caregiver interviews, comprising perspectives from 445 unique caregivers and program participants. Most interviews and caregiver/participant focus groups occurred in person, but some conversations occurred via phone to accommodate staff and participants. Table 1 provides more detail on type and purpose of key informant interviews.

<table>
<thead>
<tr>
<th>Qualitative Research Mode</th>
<th>Data Collection Timing and Sample</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document Review</td>
<td>Review of program documents</td>
<td>Gain baseline understanding of program scope and purpose</td>
</tr>
<tr>
<td>Phone Interviews</td>
<td>One semi-structured baseline phone interviews with leaders from each awarded organization.</td>
<td>Gain baseline understanding of program scope and purpose</td>
</tr>
<tr>
<td>Site Visit Interviews</td>
<td>10-15 interviews conducted during annual site visit per program. <strong>Sample:</strong> Participants of key stakeholder groups involved in awardee intervention, including lead staff, providers, informatics team, evaluation/monitoring team, workforce training team, and partner organizations</td>
<td>Gain understanding of program implementation processes, challenges, facilitators, and outcomes</td>
</tr>
<tr>
<td>Patient/Caregiver Focus Groups</td>
<td>One or two patient/caregiver focus groups conducted during annual site visits. <strong>Sample:</strong> Approximately 4-9 participants that reflect the most important clinical characteristics, demographics and payer mix of the awardee population</td>
<td>Gain understanding of program impacts and implementation from participant and caregiver perspective</td>
</tr>
<tr>
<td>Provider Focus Groups</td>
<td>One or two provider focus groups conducted during annual site visit per awardee site. <strong>Sample:</strong> Sample of approximately 9-10 providers that reflect the most important provider groups in the program</td>
<td>Gain understanding of program implementation from provider standpoint</td>
</tr>
</tbody>
</table>
Due to the ongoing feedback loop intrinsic to rapid-cycle evaluation, qualitative analysis had to progress quickly and efficiently. Over the course of two years, we generated qualitative analyses for six quarterly and two annual reports. A meta-evaluator for all 107 programs offered guidance on research domains and we used these themes to inform creation of interview protocols, reports, and qualitative coding (Berry et al. 2013; CMS 2014). The diversity of programs created challenges around developing qualitative analytic tools that could be applied across all interventions, as we describe in more detail, later in this paper. Quantitative and qualitative teams conferred throughout the evaluation to exchange information on quantitative program impacts (e.g., utilization) and qualitative impacts (e.g., changes in quality of care) and identify areas where findings were aligned or explore and findings that did not align. These discussions helped us guide protocol development for interviews or program-specific analyses in reporting.

Choosing an analytic approach

Although there are classic approaches to qualitative analysis, we benefitted by remaining open-minded about adapting best practices to suit rapid-cycle timeframes. Ultimately, we leveraged multiple methodological strategies to make qualitative data analysis both rigorous and agile. Our coding process was informed by a pre-existing cross-program analytic framework (Berry et al. 2013; CMS 2014) as well as themes that emerged from our transcripts and notes. Below, we describe some of the considerations under inductive and deductive analytic approaches, as well as the combined approach we ultimately followed, which may help inform other rapid-cycle evaluations. While multiple accepted approaches to qualitative coding (e.g., iterative review, concept analysis) support different types of study designs (Greenlagh 2010; Sofaer 1999; Yardley 2000; Morse and Mitcham 2002), there are two commonly used approaches to qualitative coding: inductive and deductive.

I. Inductive coding

Inductive coding is an exploratory approach (Stebbins 2008) that refers to using prediction or forecasting, especially based on the facts already known, to create a series of hypotheses (Thomas, 2006; Benaquisto 2008). Essentially, inductive research works from the ground up and allows researchers to work without constraints of preconceived results and encourages findings to emerge from “the frequent, dominant, or significant themes inherent in raw data” (Strauss and Corbin 1990; Thomas, 2006; Bradley, Curry and

---

2Quarterly report focus on qualitative or quantitative data was dictated by a combination of factors, such as interests of CMMI, availability of qualitative data, and site visit timing (in some cases, data was not yet collected).
Devers 2007). Though typically desired, a purely inductive approach would have required significant time and effort to identify themes among the 18 HCIA programs, especially given the volume of data we obtained and different types of interviews we conducted. Further, an inductive approach would suggest creating a codebook only after all data were collected to ensure that the team identified all themes present in the data. However, this was not feasible given our interim report deadlines and multiple round of sites visits. Additionally, inductive coding and analysis does not necessarily include causes, explanations or relationships among people or speakers (Thomas, 2006), which were themes of interest to us.

II. Deductive coding

Deductive coding reflects a top-down approach to coding that involves verification of themes (Stebbins 2008); researchers identify a theory for the topic of interest, hypothesize, observe and then confirm (Yardley and Marks 2004). This type of coding is beneficial if researchers want to apply “existing theoretical ideas” to a dataset (Yardley and Marks 2004), which they then synthesize into a list of codes (Saldaña 2015). A purely deductive approach was unsuitable for this evaluation because the cross-program analytic framework we derived from the meta-evaluator and implementation contractor (Berry et al. 2013; CMS 2014) was primarily conceptual and high-level and reviewing captured data inductively would help us identify more granular codes. On one hand, deductive coding can improve reliability because the pre-determined codes allow for higher agreement between the coding and the study’s framework or goals (Thomas 2006; Saldaña 2015), but on the other hand, coders can force data into codes when it may be appropriate to generate new codes that were not part of a research’s initial framework; this ultimately limits researchers’ ability to identify and develop new findings (Huberman and Miles 1994; Bradley, Curry, and Devers 2007). To avoid these pitfalls, we leveraged the benefits of both inductive and deductive approaches.

III. Combined approaches

A combined approach enabled us to verify pre-identified themes (Stebbins 2008) and offered the benefit of focusing on both a theoretical framework and themes detected from observations in interview data (Garrison et al 2006; Bradley, Curry, and Devers 2007). After deductively developing codes based on our research questions, we used inductive transcript analysis to refine definitions as well as determine whether to add or delete codes. Hybrid or combined inductive/deductive approaches have also been noted for flexibility and adaptability to various settings or materials that may be coded (Fereday and Muir-Cochrane 2006; Garrison et al. 2006). This advantage was especially useful given our diverse portfolio of 18 programs.
A combined approach, however, can introduce new complications through its increased complexity, such as high inconsistency and low reliability among coders, therefore lowering overall replicability of the coded data (Garrison et al 2006). For this reason, we instituted rigorous coder training. Below, we discuss our combined approach in greater detail, followed by our training methods.

**Codebook Development, Coder Training, and Refinement of Themes**

In line with best practices, codebook development was an iterative process, and we continually triangulated and refined themes. Codes were grouped under larger umbrellas for related themes: “code families.” Given the diversity of HCIA programs that participants were exposed to, the need for developing cross-program findings, and pressure to report on a continual basis, we adopted an initially deductive codebook and then used inductive methods to refine and finish shaping the final product. The specific order of inductive and deductive processes saved us significant time by simultaneously beginning codebook development and coder training, and also continuing definition refinement as coders applied them to the transcripts and notes. This ensured that we captured all major themes of interest from our qualitative data set.

Figure 1 shows the final iteration of the codebook contained a total of 28 codes and 19 sub-codes (under codes with an *) housed under the following six families: cross-cutting challenges and facilitators; program components; implementation effectiveness; program effectiveness; workforce; and endogenous/exogenous context. Staff and participants typically provided open-ended responses. Therefore, we instructed coders to apply multiple codes to the same text as necessary. Sub-codes helped us add more granular themes to the codebook, such as types of care coordination activities; coders also conceptually grasped sub-codes more easily than learning new codes, which can seem intimidating while learning a longer codebook.
In order to analyze the 18 diverse programs in this evaluation systematically, we deductively generated codes and code families based on cross-program research questions derived from an implementation contractor and meta-evaluator (Berry et al. 2013; CMS 2014), such as, “What were the key characteristics of the awardee team that would affect implementation of the innovation?” and “To what extent did the policy and political environment support or conflict with implementation?” Through this first approach, we conserved the time and resources it would have taken to derive codes inductively from the interview data across the 18 programs. The meta-evaluator framework also shaped the project’s semi-structured interview guides, which ensured that codes would correspond with interview data, and this helped reduce likelihood that coders would force data into suboptimal codes.

Next, we used inductive transcript analysis during coder training to identify new codes, delete unused codes, refine code definitions, and group themes by using the constant comparison method wherein coders compare their results over multiple rounds of practice.
(Fereday and Muir-Cochrane 2006; Glaser and Strauss 1967). Figure 2 shows the cycle of coding, IRR calculations, consensus discussions, and codebook refinements.

For example, in early stages, coders noted that program staff tended to conflate and intertwine program history and program driver themes in interviews. Consequently, we combined these codes into one because we could not develop consistent rules on how to disentangle the discussion threads. This review of the primary data also revealed whether there was a need for additional codes, such as a health care provider engagement code. We also created sub-codes under larger concepts to help coders classify information accurately and added “other” codes to capture information that did not seem to fit within the current coding scheme so we could visit data under the “other” codes at will and determine if additional codes were needed.

**Training a large team**

Given the evaluation’s volume of data, a team of five to seven coders was the most feasible approach to analysis. Multiple experts agree that the coding process should involve a team of researchers with differing backgrounds to improve the breadth and depth of analysis and subsequent findings (Denzin 1978; Mays and Pope 1995; Patton 1999; Pope, Ziebland, and Mays 2000). To the extent possible, coders cleaned and coded transcripts from site visits they had attended. This strategy leveraged coder expertise, as background knowledge of the program enhanced coders’ ability to interpret complex discussions and improved the team’s shared understanding of concepts via consensus-building discussions. Importantly, harnessing pre-existing staff expertise conserved the time and resources it would require for coders to become familiar with programs.

We cross-trained coders to establish reliability then moved into independent coding; this was more time-efficient than an alternate approach wherein multiple coders code and recode the same document as a quality assurance measure rather than establish IRR (Bradley, Curry, and Devers 2007). Throughout a four-week training, we calculated IRR with Cohen’s Kappa on independently-coded transcripts using a function in NVivo. We set an IRR benchmark at 70%, which falls in the middle of the range of Kappa scores considered “substantial agreement” (61-80%); IRR from 81-100 constitutes almost perfect agreement (Viera and Garret 2005; Bakeman 2000). By regularly conducting consensus-building discussions.

---


4 We used a combination of transcription companies and highly-detailed notes to develop codable documents. Coders cleaned these transcripts and detailed notes by listening to recordings of the original interviews.
discussions about how to interpret concepts identified in transcripts, we built and retained target IRR (Glaser and Strauss 1967). We compared the IRR of each member against every other team member at least twice during training; these calculations enabled us to identify disparities between specific people so we could identify who needed additional training and support and track which coders were successfully attaining desired level of IRR.

We took into account different experience levels across the coding team and worked to ensure that all coders shared a common coding sensibility. To account for differences in programmatic knowledge, coders who had expertise related to the practice transcripts added annotations to the documents and distributed them to the team before coding commenced. These annotations did not instruct the team how to code but merely added clarifying information. Together, we developed a harmonized sense of judgment and all coders reached at least 72 percent IRR and 95 percent agreement with at least two other coders. By the end of training, each team member had coded over 100 pages total. From this point, coders independently coded transcripts that corresponded with programs within their range of expertise. Training and our first round of coding took approximately 13 weeks. As team members attended more site visits, their programmatic knowledge grew, which enhanced their ability to understand codes and propose important code definition refinements.

We leveraged our knowledge of gaps and trends in the quantitative data in refining codes and definitions, creating a tool to complement and balance the quantitative analyses.

Cross-coding and code complexity

We used a mixture of concrete programmatic codes and conceptual codes that were more abstract and thematic. Program component and workforce codes were mainly descriptive and easier to apply; they got at the “who” and “what” aspects of intervention activities. For instance, we instructed the team to place descriptions of any activities performed by clinicians or lay health workers under their respective codes and capture any discussion of program components under the appropriate code within that family. We also created valenced codes, which capture a positive or negative direction (e.g., challenges; facilitators) that overlaid other codes. As a result, we employed a lot of “cross-coding,” meaning that we captured the same section of text under different codes. While this increased the volume of data under some codes, cross-coding and the ability to locate cross-coded data multiplied the number of themes we could search for and added specificity to the range of questions we could answer with the data. For example, rather

5 Approximately six months later, after a second round of qualitative data collection, coders replicated major elements of the training outlined above until the group reached 70 percent or higher inter-rater reliability.
than simply reviewing all data coded under “home visits” to identify challenges, we could pull up data that was coded under the both “home visits” and “challenges” to hone in on specific information (see “Electronic coding via NVivo software”).

Due to the themes emphasized in our interview protocols (and consequently reflected in interview transcripts), we created some conceptually complex codes such as program replicability and dosage (how programs assessed how much of the intervention participants received) and arrived at a common understanding of what constituted challenges and facilitators. More complex codes required a higher degree of coder judgment and interpretation. When a program leader, for instance, described implementation differences across multiple sites, coders applied the replicability code even if the speaker was not responding to a direct question about replicability and had not framed their answer using that specific theme. To address challenges arriving at coder agreement, we ensured that codes had precise decision rules—which we periodically refined based on coder feedback, shared real examples from transcripts with the coding team, and built specific exclusion criteria into the codebook. We also recommended that coders use interview questions captured in the notes and transcripts as possible flags for which codes to apply, though the coding team recognized that respondents could stray from the discussion topic and introduce additional themes.

**Quality assurance**

We used multiple mechanisms to monitor coding quality. Throughout independent coding, we resolved questions through consensus-gathering discussions with one or more team members or between a coder and senior staff working on the evaluation. Nine weeks into independent coding, the team participated in one additional round of IRR calculations to ensure that reliability remained high. An experienced team member also randomly spot-checked finished transcripts and managed coding questions to verify that decisions were consistent. We periodically tested coder judgment by sending out short challenges and escalated important questions from other coders to the entire team or to program experts. The larger evaluation team added another layer of quality assurance to interpreting coded data by reviewing findings and clarifying nuances or lending their program-specific expertise to validate hypotheses.
Adapting to Rapid-Cycle Analysis and Reporting

Given the demands of high volume of data and compressed timeframe, we utilized a number of strategies to facilitate analysis.

**Identifying research priorities**

Transcript prioritization helped the team organize workloads and enabled us to access coded themes across all programs on a rolling basis. For example, we identified lay health worker roles as a theme of particular interest in the beginning of the evaluation and instructed coders to code all lay health worker interviews across the 18 programs before moving onto other types of intervention staff. Later in the evaluation, we coded all participant interviews, which matched up with delivery of quantitative data as programs closed out under their awards. As a result, our knowledge of gaps and trends in the quantitative data enabled us to refine codes and definitions to complement these insights, whether through corroborating or explaining differences between qualitative and quantitative results.

**Data preparation**

In order to respond to the needs of a rapid-cycle evaluation, we changed our data preparation approach during the evaluation. We realized that transcript cleaning was unsustainable and determined that coding highly detailed notes was more efficient to reduce the amount of data we had to analyze (McLellan et al. 2003). In the first round of data collection (April – October 2014), we sent anonymized recordings to professional transcription companies; however, staff spent multiple hours per transcript revisiting the interview recordings to correct errors and fix medical terminology. Given the short turnaround times of a rapid-cycle evaluation, the time dedicated to data preparation became a barrier to analysis. Therefore, in the second round of site visits (February-June 2015), we instructed note-takers to capture all thoughts expressed in each interview and to compare interview recordings with the detailed notes as a quality assurance measure. Staff members were able to take notes that were nearly verbatim and condensed language when necessary. Avoiding the redundancies of cleaning professional transcripts, taking detailed notes sped up the coding process without losing the level of nuance provided by transcript analysis.

**Use of computer-assisted qualitative data analysis software**

Computer-assisted qualitative data analysis software (CAQDAS) facilitates review and sorting of large volumes of data. For this evaluation, we used NVivo (QSR International Pty Ltd. Version 10, 2012).
NVivo can run queries to locate data captured under a single code, but this program was particularly useful because it can locate cross-coded data, which helped us quickly access information on granular themes. Using NVivo also simplified training. NVivo can quickly calculate IRR between two coders using Cohen’s Kappa both at a global level and per-code. Using these data, we could hone in on pairs that had higher levels of disagreement to tailor training and look for trends across pairs at the code level (e.g., specific codes that were particularly difficult to apply with high reliability).

**Staging coding**

In order to deliver results in a timely way, we scheduled three separate rounds of theme-based coding using the same set of seven trained coders. Because coders also attended site visits, staff capacity for coding was higher after each round of visits and program-specific knowledge was fresh in the team’s mind. We began by coding frontline staff, leadership, and partner interviews after each round of site visits. As programs began to report on quantitative outcomes, emphasis shifted from implementation effectiveness to program effectiveness and in a final third round, we coded all patient and caregiver focus groups; these interviews in particular contained themes about patient outcomes and program effectiveness. Since this was later in the evaluation, we had a better understanding of outcome-related themes and what analyses would help explain or add nuance to quantitative data.

**Discussion**

Through a descriptive analysis of the strategies that the project team undertook to analyze 18 disease-specific programs under the CMMI’s Health Care Innovation Awards, this paper has described an adaptive coding methodology. The role of CMMI in testing innovative care and payment models informs a broader scope of policy and government-supported programs to promote health equity and mitigate health disparities. The mixed-methods approach of the evaluation allowed qualitative analysis to help explain and add nuance to quantitative findings in order to interpret intervention effectiveness and identify components that can be replicated to help wider populations. To do this, it was imperative that qualitative analysis was both rigorous and time-efficient so that CMMI and interventions in the portfolio could access a real-time story of programs and their impact as interventions progressed.

We present top lessons learned from our experience evaluating the 18 HCIA programs, although lessons reflect methodological aspects of analysts that may apply to other sectors or evaluations that involve a high volume of data and compressed time-frames. These recommendations are in the approximate order of what to consider when planning qualitative coding.
1. **Determine whether an inductive, deductive, or combined approach is appropriate.** There is no one “right” approach to qualitative data analysis. As we found in the HCIA evaluation, some methodological choices were governed by the relationships between reporting requirements, logistics of data collection, and a continuous evaluation cycle.

2. **Consider the purpose of different types of codes and how to use them effectively.** Multiple “types” of codes may appear in a single codebook. Simpler and fewer codes will enable staff to learn coding schemes quickly, however, oversimplified code families can result in data that is not specific enough to generate findings. Though we often reviewed data under a single code, cross-coding, facilitated by NVivo, enabled us to locate text that had been coded under two ideas; this generated lower volume, more specific data.

3. **Identify the research needs and purpose of the project early on and design analysis around them.** Consider logistics such as data collection or deliverable timing that may shape the pace or order of data analysis. Prioritize themes as necessary.

4. **Weigh the costs and benefits of coding over multiple rounds or in one final round.** Due to the rapid-cycle design of this evaluation, multiple rounds of coding enabled us to generate validated findings throughout the entire evaluation process. Coding in multiple rounds enabled our team to refine code definitions continually and enhanced the team’s collective conceptual knowledge. Conversely, coding in fewer rounds later in an evaluation may be beneficial for some teams because themes of interest will have been identified, solidified, and refined.

5. **Consider staff resources as they relate to other rapid-cycle evaluation activities when determining the size and composition of coding teams.** We used a large team composed of program experts that was ultimately beneficial despite the time investment of training multiple coders. Having a larger group at our disposal mitigated challenges such as short reporting deadlines; a large volume of data; multiple rounds of coding conducted at different times; and the prospect of staff turnover.

6. **Leverage staff knowledge to enhance reliability and use time efficiently.** Although codebooks should be created so that any individual can theoretically apply the codes with high reliability, independent of their background knowledge, we encountered challenges that arose from complex interview discussions. Due to the number and complexity of programs in the HCIA evaluation, it was helpful to use program experts as coders, especially because interviewees could be vague or refer to internal program information that non-experts would find confusing.

It is vitally important to consider the context of a coding project, such as coordination between coding and data collection related to staff capacity, development and refinement of ideas and themes as evaluations
progress, timing of expected deliverables, and volume of data collected. This context determines many of the factors in designing and implementing a codebook, especially in a client-proposed evaluation setting.

Given the growing public and professional interest in health services research and publication, it is important to remain open-minded in the application of traditional qualitative coding methods in innovative ways. Through applying qualitative coding to a rapid-cycle evaluation structure, we have shown the adaptability and flexibility of the strategy while remaining consistent with rigorous methods. Moving forward, more insight and dissemination on practical strategies in various contextual influences will help frame the further development of qualitative coding.

### References


