The second thing that I want to say by way of transition is that process of having – developing a set of instructions, having two coders read independently every answer, negotiate out their disagreements after reaching sufficiently high agreement rates, took us, pause, four years. So it’s 2012 and we’re just releasing now the codes of the 2008 data. That’s how embarrassing this is.

We had lots of effort, lots of false starts, lots of problems, and lots of examples of low levels of agreement until we finally got the whole thing working. Now, hopefully what we’ve done in figuring all this out would go more quickly the second time around with human coders. But, the next set of presentations explore the potential approach to make it much more efficient.

That instead of having human coders code 2,200 responses to every question, could you have human coders code a small number of the responses, and then use that coding to train a computer to do the coding the rest of the time for you? Is it straightforward enough to do that? And so natural language processing is a field that came up a lot in the conference that Skip alluded to that we organized at the University of Michigan some years ago, where we brought lots of experts together, including experts from the firm called Language Logic.

So, Rick, should we call it “Ascribe” or “Language Logic”? [Response from audience]. Okay. So the company’s name is Language Logic. And so Language Logic was one of the organization’s there. There was a gentleman named Fabrizio Sebastiani, who’s an Italian scholar, who had, at that point, written some software to do this kind of coding and developed an algorithm, which has been a – and Ascribe is the software that implements his - [response from audience].

So since we completed the human coding a little while ago, we’re now gonna welcome Dean Kotchka upfront here so that he can tell people quickly about the initial foray into can we use just the first set of human coding, and teach a machine to do the rest and have the machine finish it up. And after lunch, Mark Leiberman is gonna talk much more broadly about the potential of natural language processing and computers to do this kind of coding.

**Dean Kotchka:** Thank you. Thank you all for squeezing us in. We are not part of the agenda that has been published normally. Yes, we are now, so I can take as much time as I want. No, I can't – I can't take – but I do have a few minutes and Jon and Skip did want me to point out a
couple of things that we’ve done subsequent to the coding work that’s been described.

I also might point out that we didn’t make any money on this job that gone over a four-year period of time, but we also learned an awful lot. And so with that, let me get started. I also have to say this is – I’ve given a lot of presentations, but this is a very intimidating thing because I’ve not been before with a scholarly group such as this presenting. So I really do appreciate the opportunity. I’m probably not presenting it in a way you typically might see a paper being done.

I’m also going to go through it fairly quickly because I don’t want to stand between you and lunch, nor any question times that you might have for Skip’s presentation. I do want to take a minute though since I’m not a researcher, and since I’m not in academia or have any published work, at least run down – a few seconds to run down what Language Logic is and who we are to maybe provide at least some assemblance of credibility as to why I’m standing up here in front of you.

We were founded in 1999, and I am gonna go through these fast. We published a verbatim management software system called Ascribe. I use the term “verbatim management” because in addition to coding, which is its principle functionality, it’s also used for transcription and translation work. We’re used by over half of the Honomichl 50 research firms. For those that may not know Honomichl, it’s an annual listing of the top market research firms in the world organized by revenue.

So we have a pretty good footprint embedded in these company systems. We have about 12,000 users in 56 different countries. This year, we’ll process about 300 million verbatim comments through our application. And in addition to selling the software and licensing it, we have a solutions group, which uses our own software to provide coding services and translating services. That’s the group that the ANES people worked with to get the coding work done, and we had two dedicated people working specifically on this project.

It was an interesting exercise for us because it tended – because of some of the issues about reliability and the need for documentation and good procedures and so forth, tends to go against the way we work. We have always had encouraged coders to talk back and forth as coding was going on. In this case, as you heard, it was specifically intended to not have that, to keep a far wall between
them until such point as the instructions were determined to be understood, and then you got into the reconciliation process.

We have about 100 full-time contractors scattered out through the world doing this work on our behalf. And this group, itself, will process about 20 million comments this year. I’m gonna skip through these because this has already been discussed. The open-end dilemma of balancing the various costs and factors against needs for consistency and accuracy speed, and so forth.

I will point out that we have seen over the last couple of years, in addition to being a very labor-intensive cost element of open-ended coding, speed is becoming increasingly more important, at least to our users. Since data collection times have been compressed via Internet and mobile-type surveys, the time that people are asking for results to be processed is also being compressed down.

So if it took two weeks to collect the data, it was reasonable for somebody to take an additional week or so to tabulate it. But now if the data is being collected in two days or one day, then the coding groups, the operations groups in these firms are being asked to turn it around.

And in addition to that, we’re seeing an increase rise in what I call tactile surveys, one/two question, one open-end, two closed ends, that go out and response to a Twitter spike somewhere, where they are trying to find and narrow down some specific information. Those kinds of surveys are also increasing the need for speed. We have been working for the last couple of years, as it was alluded to, to building some answer to this, to continue to provide a reasonable degree of accuracy, consistency, reliability, but yet improve the speed and reduce the cost.

And the solution that we have built into our software, I’m referring to it generically as an automated coding model that’s not our brand name. And we built it in about two years ago. It’s a “learn by example” process. You feed it trained, coded examples, and process it and build binary classifiers, and then use that list of classifiers to code additional data.

As Jon mentioned, it’s based on the work of Dr. Fabrizio Sebastiani. He’s with the Institute of Information and Science Technology based in Pisa, Italy. He was, I think, at this conference – a similar conference in 2008 in Ann Arbor. And we have licensed their technology and built into our platform. I have some
published papers from Dr. Sebastiani. I was gonna put links in there, but realized that those links are embedded in our software and you’d have to have credentials to get to them.

But I’ll be happy if you want to exchange a business card, I’d be happy to get you the information and the links to get those papers. Or you could probably just Google him and find them as well. This I’m going to run through fairly quickly. Basically it’s just a systematic of how the automated coding model works. You start with uncoded verbatims, and a human coder codes those. We consider those training examples.

Binary classifiers are created, and then the process is used to code/uncode the verbatims. A cycle – it’s an iterative process as you review some of the results and resubmits them to the model. I’m gonna go through this again pretty quickly. I’ve got the five-minute note. What we decided is that if we were sitting on a ton of coded data that had been vetted through a very rigorous process, and one of the big problems with an automated coding model is if you put ambiguous or bad data in, your model is not going to perform very well.

So we took it upon ourselves to take one of the questions – we actually selected two, but it’s essentially the same question, a dislikes question, republican and democrat. The question text is: “Is there anything in particular you dislike about the Republican Party?” I don’t know if you can see this or not, the codebook. This was the code frame that was used to code the information. I put it up here only simply to point out that this is the same code frame, that there are a couple of categories.

One up there: general. That’s a very hard – the general category is a very hard category to give specific examples about, so that may or may not be a good working code in the model. When I put up that on the plane today coming in, I was wishing that I had more than a couple hours to talk. But let me run through here. Reconciled comments: That’s the number of the comments that were coded by the coding team for Skip and John.

Of those, we selected 1,489 to be training examples, that’s what we used to train the model. The remaining 1,425, we pushed through the model to be coded. So there are the 1,425 that we pushed through the model. Of the 1,425, the model coded 1,137 of them. When we set the parameters and the knobs and the dials of the model, we can establish a confidence level, below which we don’t want any codes applies, above which will apply a code.
So, in this case, a few hundred of them didn’t have any codes at all applied. Seven hundred and thirty-seven of the 1,137 were coded identically. But, really, we look at the results at the code level, there were 1,734 codes applied to that base, 1,425 by the human coders. The ACM, the model code applied 1,618 codes. The model applied 271 codes that were not applied by the base coders.

In our software, we don’t call it precision; we call it over coding and under coding, but it similarly – it missed 368 codes. We calculated an overall accuracy percentage at 84 percent. This – again, I don’t know if you can read this. This is my final slide and just a shot of some of the differences. This is obviously the verbatim. The second column is the model coding, and the third column is the human coding. The blue indicates codes that the coders put in; red indicates codes that the model put in.

I guess the conclusion here is that there really isn’t a conclusion other than I think it’s at least encouraging that – and this had no iterative update process and this was just done by our coding group. And so I think there are some opportunities – we are not researchers, but there are some opportunities to formulize an experiment, a good experimental design. I think there’s an opportunity to further refine the model by building in some of the errors that were found.

And then also to apply it across all the questions that share that same codebook because you would think that everything that is gonna get coded in that codebook, and that codebook has had that set of examples built for it. It’d be interesting to see how it runs against all the data. So I hope I didn’t go too far over my time. Thank you.

**Jon Krosnick:** Okay. So I think we’ll do ten minutes for questions either for Dean, who is leaving already, or for Skip, who I hope is still there. Can we put Skip on the screen, Dave? There he is. Okay, so question/comments about coding?

**Male:** Just a small question about coding the knowledge of political office. Since you said earlier, at the beginning, that people use these to access general knowledge in politics, why did you decide to exclude the general knowledge information that was contained in the answers? So you said you coded it correct, only the ones that get the job right, whereas if the purpose of the question was to access general political knowledge, then, presumably, you’d want to communicate that in the codes.
Skip Lupia: So it’s a great question. The problem with the question that’s written is that’s actually not designed to assess general knowledge. A question that was assigned to address general knowledge would ask something like: “Tell us everything you know about Person X.” But that question doesn’t ask that. It says: “Do you know what job or political office this person holds?”

And so to use it as a source of information about what general recall an individual has about a candidate, really, it has to be skewed because to really get an unbiased sample of general knowledge, the respondents would have to systematically choose not to answer the question. And they’d have to choose not to answer the question we asked in the same way.

And so ultimately what we decided to do was code answers with respect to the question that we asked. I mean my sense is like yours that I think most people want to use this as a general knowledge question, and this is not a good general knowledge question.

Male: And so did you consider adding a second code for the questions that said – got the answer wrong for the question, but indicated some knowledge of the subject?

Skip Lupia: So we’re releasing two sets of codes. So this is – in the past, we used to just – for every question, there would be one code and it would have two values: correct or incorrect. So now we have a first initial set of codes, which basically just characterize the answer. So there’s a variable. Did they say something correct about the political office? There’s a variable. Did they say something correct about the job?

And then what we’re manufacturing are aggregations of those, which did they say anything correct about the political office or the job. And I think the distinctions that we’re gonna use is did they say anything correct and nothing incorrect. Did they say something correct and something incorrect, or did they say nothing of the kind? And so if I say that William Rehnquist is a bozo, I get put in “other”, and we don’t judge the truth value of that claim.

Male: So, Jon, I’m a dislikes quotation – likes/dislikes. The way it’s done now, could one replicate what congress did on levels of conceptualization back when this question was used in the sixties?
Jon Krosnick: Well, first of all, we can't replicate what congress did in the sixties. So the idea of trying to somehow map what we’ve done back into those earlier categories was – just so people know what we’re talking about here; so levels of conceptualization were inferred for people by placing them into a set of buckets. If they were at the highest level, they were ideological thinkers, so they were the words “liberal” and “conservative” in ways that showed that they understood what they meant.

One of the lower levels was group thinking, so talking about how this candidate is good for the working man. So it would be very different – you would have to implement a different coding exercise in order to produce those, I think. You’d have to define what qualifies as ideological, what qualifies as group, and so on. And as you know, those categories were treated as mutually exclusive.

So a respondent was placed either in the ideological bucket or in the group’s bucket or some other bucket, and respondents give combinations of those instances. Probably if we were to develop a new set of coding instructions, people could mention group things and ideological things, and we would represent that.

Male: Yeah, I think you guys had to draw the line between digesting this stuff for the users, saying this is correct and this is incorrect, versus giving them the verbatims and saying you do what you like. I mean you’re sort of trying to strike a balance, I guess, or some way of getting the benefit of both of those approaches in some ways. But it’s kind of like the AAPOR standard definitions. You could report a response rate, defined as a AAPOR response rate three.

Or you could provide people with all of the outcome codes and have them figure out whether what you got was – using a different rule, what you’ve might have gotten. So I’m wondering how did you come to the balance that you wanted to?

John Krosnick: Well, we tried to get away with the simple thing first. We tried to put out all the verbatims and leave it at that and let people do whatever they wanted. But it became really clear that people didn’t want to have to deal with that, but they want codes. So then in order to produce codes, I mean you heard – Skip and I bring different perspectives and strengths and values to what we did.

And one of the focuses for him in our collaboration has been principle basis for what we provide to people, rather that just saying let’s find some buckets. So a bunch of people said,
“Unemployment; let’s create an unemployment bucket,” which felt impossible to implement comparably across multiple surveys. So we struggled – based on partly advice from user committees of people who actually used these data in print.

And this is maybe one of the most striking things of all; I don’t think Skip said the number of like/dislike categories. There were – do you remember? There were like 500, 700, or some gigantic number of categories. So we gathered up every publication we could find, and nobody used the 700 categories. They had all combined them down into like five.

And so the question is why are the coders struggling to create 700 distinctions when users really only want to know did he like the personality, did he like the foreign policy, and did he like the performance history, and so on. So we were informed partly also by what users seem to want to do in the past. But there is no doubt we didn’t – what we’ve come up with is only one possible approach, and there would be others that would be legitimate.

**Male:** So in some cases, you used an inductive approach looking at the responses that came over the transom. And then in the budget case, you imposed an external template. So I’m wondering how did you decide those two different approaches?

**Skip Lupia:** I was gonna say we tried to go with an “x” anti set of categories first, and likes/dislikes, we just struggled. It was a black hole.

**John Krosnick:** That’s the answer.

**Male:** Yeah this is slightly related. Because I’m thinking if I were doing an open-ended question, the alternative is a closed-ended question. And it would be nice to have a close-ended question covering somewhat the same space as the open-ended, and then you’ve got a check, and then the other is with a focus group. Because you went to the U.S. budget thing, but if you had some focus group people, you might get a better sense particularly of the ambiguous things.

So people are talking about Rehnquist, or whomever it is, and you quickly understand, yeah, the guy who calls him a bozo, he knows what he does. And that would be an alternative way of – sometimes matching the open-ended with the coding with now close-ended or focus group, you could go and ask somebody in the focus group afterwards. Or you could have them answer your open-ended, and then you query them afterwards and get some other measures.
Yeah, just to draw a link with what Collin said earlier and with what you’re saying now, one of the things that we discovered was that when – we have the audio recordings of these conversations between the interview and the respondent, and tomorrow, Hector Santa Cruz is gonna talk about what he discovered from those open ended – those recordings. But one of the things that we found were instances where the question is: “What job or political office does Dick Chaney hold?”

And then the respondent says: “the devil incarnate”. And then the interviewer says, “Well, I have to record an answer here, what answer do you want me to record?” And the response is, “The devil incarnate.” And so the problem is that we haven’t told the respondent the purpose of the question. That if we told the respondent that the researchers want to know if you actually know the job – not how you evaluate him, but if you know the job or political office this guy holds, could you also tell me that as best you know?

And if the respondent then said, “Oh, he’s vice president, and I wish he wasn’t,” or something of that sort. So it raises a really interesting insight about the idea that maybe the interviewer does know. And wait until you hear what Hector has to say about that tomorrow. And that certainly the respondent doesn’t know what the purpose of the question is, and they may handicap themselves.

But I think to come back to your question about if we’re trying to measure general political knowledge, the notion of these open-ended questions about jobs or political offices is typically, and I think now being viewed from an IRT framework, that is this is a question that has a right answer, and it’s a difficult one or an easy one. And so it turns out that lots more people know who Dick Chaney was than know who John Roberts was.

And so if you know a right answer and you’re getting a certain percent correct, you can build a battery of items on lots of different topics. If you rephrase the questions as do you know anything – “John Roberts works in the federal government. Do you happen to know anything about him?” And imagine somebody says, “He’s a male.” Do you – is that a right answer, or did you sort of give them that by the name?

So that’s a little silly illustration of the problem we ran into where when we thought about trying to create a category that represented
anything they said that might be correct, you start getting off into some silly territory.

*Male:* My point really was given you did an awful lot of work on these open-ended, so you have an assessment of whether they contain knowledge like liberal devil anti-Christ, which suggest an intimate understanding of American politics, so that you could give people information that then don’t have to replicate themselves. So it was really not so much question of what you did wrong, or that you’ve done more work that that, and it would be generous to share that work with users so that they could take advantage of it, and then perhaps rethink the question for the next time.

*Male:* Using the IRT model, you might want to think in terms of a partial credit model. So the person says that he’s on the Supreme Court, but they don’t say Supreme Court, you know, the chief justice of the Supreme Court, then characterize that has a partial credit. And you use kind of that approach. But to simply say they are incorrect. It’s either correct or incorrect. It seems that you really do abandon some information that’s a response.

*Jon Krosnick:* You’re validating what we did. So in other words, the old version was what you just critiqued, the correct or incorrect with a very strict criterion. And then in the codes we’re releasing, there is completely concrete, and then there’s partially correct, and then there’s other kinds to dissemble and people can recover those.

*Male:* Jon, I’ll go back to the process that was described for developing coding that started with the theoretical framework and then ended up with a reliable coding procedure. It struck me as – and this may not be new to anyone, but is very similar to what happens in item development for educational testing. The National Assessment for Educational Progress, for example, had spent a lot of years and a lot of time doing very similar kinds of work to develop items that can then be coded that can again start from a theoretical framework.

And then end up with a set of procedures that can be coded accurately for these questions that have a right or wrong answer or degrees of correctness. And there’s a literature that that you may already be familiar with, but I thought that I would mention that there has been a lot of work done on that kind of processes.

*Jon Krosnick:* Skip, do you want to say something?
Skip Lupia: Yeah, just when we were talking about the chief – Jon, it’s this other punch line about the chief justice question, which is – I don't know if you know this, but the correct title of this constitutional office is not chief justice of the Supreme Court; it’s chief justice of the United States, and that’s what’s in the –. So if you had said that, which is actually its constitutional title, you would have been wrong.

Male: I want to come back to the issue of documentation, which Skip said is so very important, and I think we all agree. But I want to gee you just an instance of it’s even broader than I think maybe what you’ve been thinking about. And it’s from an example that my mentor, Howard Shuman, wrote about. He did a study with Dudley Duncan in the Detroit area study in 1971. This survey has been repeated every year.

And so in ’71, they used it as an opportunity to repeat – I’m sorry, it’s a survey done every year, but on a different topic. And in ’71, they decided to repeat modules from preceding years. And one of the modules has been about gender roles, and they had been done in the early fifties, and so this is twenty years later. And they had an open question and they discovered some – and I can't recall exactly what it was, but very large differences that made them wonder what was going on.

And so somebody hit on the idea, and I wouldn’t be surprised if it was Howard, who is really sensitive to these kinds of things, and they said, well, you know, maybe it’s not the coding instructions, but it’s the fact that the nature of the coders, that twenty years ago when a coder read this particular answer, they coded it this way. Today, in the minds of coders who live in a different kind of environment, they coded it differently.

So they could investigate that by – they still had the verbatims from the earlier study, and so they could have the 1971 coders go back and recode the information from 1952, or whatever it was, and, in fact, they discovered very different results from the earlier study. And so this is a wonderful illustration of how particularly in the National Election Studies, but any study is going to look at this over time, that one has to be concerned about this.

But it does suggest more generally, even on long shot studies that maybe one wants to record, in terms of documentation, something about the attributes of the coders. Because, after all, you could imagine at one point in time, maybe the coders are all males, just by happens there’s another – and males might – let’s say on this
gender question, might code the same set of answers somewhat differently than females might. And so information, more generally, in terms of documentation, I think is really essential.