6

A Simulation Approach to Validate Models Derived from Observational Studies

Pierre N. Robillard and Simon Labelle
Ecole Polytechnique de Montréal
Canada

1. Introduction

Protocol analysis is a suitable approach for studying cognitive behaviors. It is a psychological research method that elicits verbal reports from research participants, and is used to study the thinking process in the cognitive sciences (cognitive psychology, behavior analysis). Consequently, much of what we know in most domains and particularly in software engineering about practitioners' cognitive processes is based on researchers' use of protocol analysis (Robillard et al, 1998). Trickett & Trafton (2002) provide a comprehensive summary of practical how-to primers on protocol analysis. Hughes & Parkes (2003) provide an excellent survey of trends in the use of verbal protocol analysis in software engineering research. They found that some difficulties associated with the technique are: (i) the effort of devising a valid and reliable encoding scheme; (ii) the time-consuming nature of the encoding process; and (iii) the problem of comparing the results of researchers who have applied different encoding schemes.

Coding is a strategy used in protocol analysis to extract values for quantitative variables from qualitative data collected from observations. Human behavior is characterized by coding schemes, and statistical tools are used to derive models of the behavioral patterns emerging from the tasks under study (Ericsson & Simon, 1993).

The coding scheme defines the formalism for encoding the protocol, and should be capable of encoding the moves adequately and yet be formal enough to support quantitative analysis. A transcript entry is called a (verbal) move. The protocol transcript or verbatim account is an accurate written representation of all the moves performed by the participants during the observed session. The coding of a verbatim account is a subjective task that is very time-consuming. Consequently, the coding activities are the weak link in any study based on protocol analysis, in that two coders who perform the same move characterizations may have a different interpretation of the moves, and may not replicate exactly the same coding pattern. This behavior is natural, and it is unrealistic to expect exactly the same coding patterns from two coders, or even from the same coder.

In protocol analysis, reliability is associated with stability between coders (Baer, 1977), such stability being obtained by measuring the degree of similarity between the resulting codes. Coding is a subjective activity based on the assignment of qualitative codes to natural moves, and many approaches have been developed to evaluate the reliability of

coding activities (Carletta et al., 1997, Seaman, 1999, Foster et al, 2008). Stability could also mean that a coder codes in the same way during the entire experiment. One must verify that the coder will not introduce bias in one way or another during the various coding sessions (Medley & Mitzel, 1963). For example, in software process assessments, studies have been conducted to benchmark Cohen's Kappa index, which is a measure of inter-rater reliability (Khaled, 1999).

The traditional scientific approach to validating models derived from experimental studies is to replicate the experiments. Experimental replication validates the appropriateness and reliability of the observed data, the correctness of the statistical analysis, and the significance of the results. Difficulties in replicating observational empirical studies arise from the impossibility of using the same people carrying out the same project in the same environment. However, a qualitative replication has a looser meaning, which is that it must only preserve the conditions set forth in the theory being tested (Seeman, 1999).

What is often recommended is to analyze the reliability of an empirical study before deriving a model based on these studies. Cronbach et al. (1972) suggest the theory of generalizability, where the factors that could affect reliability are identified, controlled experiments are repeated many times with one of the factors being modified each time, and the results are statistically compared. The idea is to define the environment for supporting the generalization of the results. This approach is considered the best for estimating reliability (Berk, 1979; Mitchell, 1979). However, it is very difficult to implement, since it requires a limited environmental setup and a relatively large amount of resources (Johnson & Bolstad, 1973). Some authors also argue about the necessity for generalizability, finding this approach unduly complex (Nelson et al., 1977) and in some cases inappropriate for experiments where qualitative measurements are involved (Cooil & Rust, 1994).

The term *validity* is potentially confusing when used in protocol analysis. The distinction is between the validity of the coding scheme, or variables derived from it, and the validity of the interpretation relating content variables to their causes or consequences. To assert that a category or variable is valid is to state that there is a correspondence between the category and the abstract concept that it represents. To assert that a research result based on protocol analysis is valid is to state that the finding does not depend upon the specific scheme or coder (Weber, 1990)

In the case of the validity of the coding scheme, measurement validity is defined by its representativeness and adequacy (Kerlinger, 1973; Curtis, 1980), validity often being confirmed by human experts (Herbert & Attridge, 1975). Indeed, a detailed analysis by experts of the experimental setup may be necessary to judge the validity of the experiment. Another approach would be to measure the same phenomenon in various ways and then compare the results (Campbell & Fiske, 1959). Many studies explore means to improve inter-rater reliability and use Monte Carlo investigation to measure the effect of the number of rating scale categories on levels of inter-rater reliability (Cicchetti et al., 1985).

Most studies are looking to improve the validity of the coding scheme and the reliability of the coders who are using it. The validity of the interpretation is crucially dependent on the validity of the model that is derived from the protocol analysis, and yet there is no single established definition of model validity and validation in the modeling literature.

Building confidence in the usefulness of a model is a gradual process, dispersed throughout the methodology, starting with problem identification, the coding scheme, the reliability of the coders, and the structure of the resulting model. The ultimate objective is to establish the validity of the *structure* of the model. Accuracy of the model behavior's reproduction of real behavior is also evaluated, but this is meaningful only if we already have sufficient confidence in the structure of the model (Barnas, 1996).

Nowadays, model structures derived from protocol analysis are built from the activity of one coder, whose reliability has been tested against that of another coder on a part of the protocol. The major assumption in protocol analysis is that a reliable model structure is built by a reliable coder using a valid coding scheme. It is assumed that, if another coder had done the coding, the same model structure would emerge.

Our research work leads us to consider the validity of interpretation of the model structure resulting from protocol analysis. We found no approach in the literature to evaluate the impact of subjective coding on the model structure.

The idea behind Monte Carlo Coding Replication (MCCR) is to simulate replication of the coding in the protocol analysis. The simulation generates codes with random variation of the qualitative interpretation within an observed range of variability. This enables the researcher to determine in which qualitative parameter ranges the resulting model structure remains invariable, or to evaluate the sensitivity of the model structure to these parameters. This paper presents an original approach to measure the impact of coding variability between reliable coders on the structure of the resulting models.

The simulation procedure consists of three steps that have been added to a standard protocol analysis procedure. The steps of the simulation procedure are described in the context of a real protocol analysis project. The next section introduces the components needed for the simulation in the context of an overview of protocol analysis, which leads to the problem to be solved by MCCR. The third section, which constitutes the core of this paper, presents in detail the three steps of the simulation procedure, which are the definition of the variability matrix, the performance of the simulation procedure and the reconstruction of the model from the simulated moves. The last section illustrates the impact of coder variability on the structure of the model from a real model built by the human coder.

2. Overview of protocol analysis and problem statement heading

The objective of the research project used to illustrate MCCR was to observe the activities of software team members during technical review meetings and to use protocol analysis to derive a model of the team's behavior during the meetings. The analysis approach used to derive the model is based on Exploratory Sequential Data Analysis (ESDA) (Sanderson & Fisher, 1994), which is applicable to the analysis of systems, environments, or behavioral data, the sequential integrity of which has been preserved. More details of this study and its approach can be found in the publications of the review meeting model (D'Astous & Robillard 2002, D'Astous et al., 2004). The model at the time it was published did not take into account the results obtained subsequently from the MCCR approach presented in this chapter.

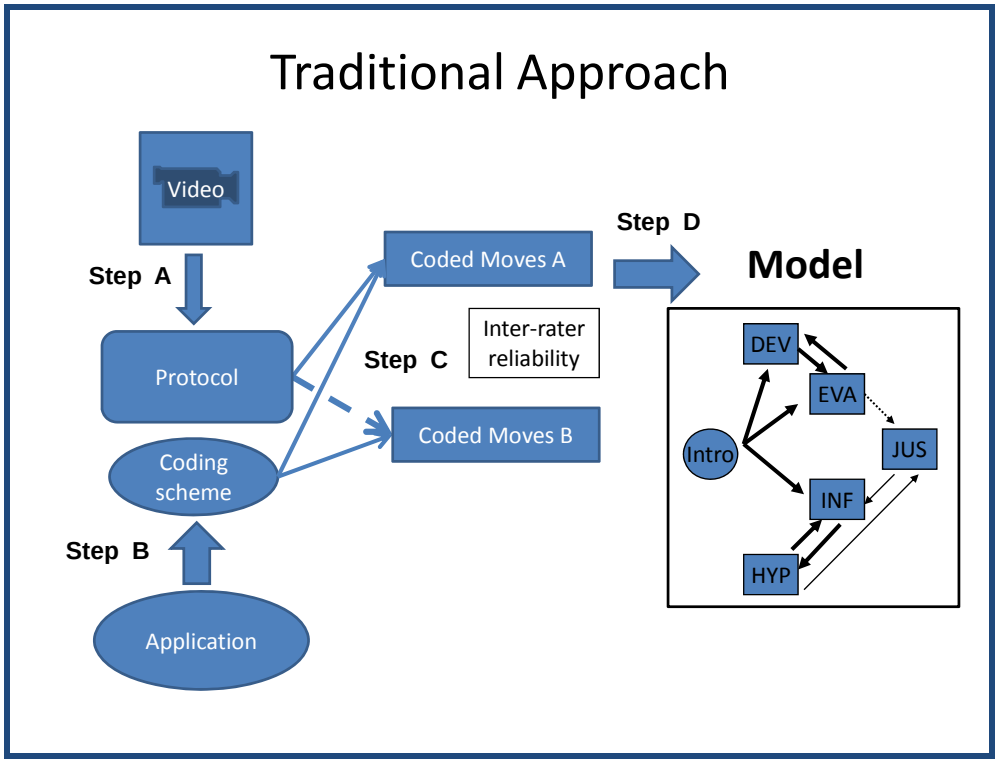

Fig. 1. The four steps of the traditional approach to protocol analysis

Figure 1 illustrates the four steps (A to D) of the traditional protocol approach to model building derived from qualitative analysis, where the data collection approach is to videotape participants. For more details on traditional approach to protocol analysis see Trickett & Trafton (2002).

2.1 STEP A: Protocol

A specially trained typist transcribes the videos to produce a document called a *protocol*. The transcription is a technical activity that requires considerable skill. The main difficulties arise from the naturalness of the discourse, the technical vocabulary, and verbal familiarity. The transcripts form the basic data for the analysis, although it is sometimes useful to refer back to the video to validate the meaning of a statement. A transcript entry is called a *(verbal)* move (Sinclair & Coulthard, 1975; Edmonson, 1981), a move being a statement made by a single speaker. A series of related moves is called a *sequence* and a series of sequences is called a *verbal interaction***.** During a conversation, a move is the contribution of a single speaker to a given sequence. It is, therefore, the basic element to be identified from the transcript. The protocol (transcription) is an accurate written representation of all the moves made by participants. Moves may be very short and monosyllabic, like No or Yes, or extended and last for a few minutes.

2.2 STEP B: Coding scheme

A coding scheme is derived from the application domain. The coding scheme defines the characteristics that are needed to describe a specific move, facilitate its identification, and provide the basis for coherent analysis. The definition of these characteristics is obtained through an iterative process of experimentation and validation of the coding scheme on one typical recording session. The coding scheme must be flexible, while at the same time maintaining the accuracy required by the researchers. The scheme should make it possible to focus on aspects judged to be relevant by the researchers. It should meet the theoretical attributes of interest and be objective (Olson et al., 1994). Objectivity deals with the reliability and validity of the coding scheme, while theoretical interest depends mainly on the domain and goals of the research. A coding scheme should be able to encode the moves adequately, and yet be formal enough to support quantitative analysis. Researchers in software engineering are even proposing a coding scheme to reduce the effort required to perform protocol analysis (Mayrhauser & Lang, 1999). Cognitive ergonomists and software engineers are often involved in coding scheme definition.

The following illustrates the coding scheme developed for the model presented in this example (D'astous et al, 2004). Many characteristics are needed to describe a specific move, facilitate its identification, and provide the basis for coherent analysis. Table 1 defines the move characteristics.

An elementary syntax, including the separator symbol (/) between each characteristic, is used to code a move in such a way that it will facilitate automatic analysis. The generic code is then:

ID/ACTIVITY/SUBJECT/DURATION/ATTRIBUTE

For example, in the 60th move by teammate M (ID), M makes an assertion consisting in the evaluation (ACTIVITY) of the design solution SOLx (SUBJECT), lasting 5 seconds and referring to the format used to describe SOLx, which is criteria F, CRIT:F (ATTRIBUTE). Its code is:

M60/EVA/ SOLx/05/CRIT:F

Table 2 defines the limited set of activities for the review sequence moves.

Characteristics	Definition
ID	Identifies the move through the speaker and the rank in the polylog
ACTIVITY	Identifies the speaker's action
SUBJECT	Identifies the informational entity on which the activity is performed
DURATION	Identifies the duration of the move in seconds
ATTRIBUTE	Complements the subject, with respect to a criterion, fact, role, content, project, sequence, etc.

Table 1. Move characteristics

Activity	Abr.	Definition
Evaluation	EVA	Judging the value of a subject. This evaluation can be negative, positive, or neutral.
Justification	JUS	Arguing or explaining the rationale for a certain choice. It is often necessary to follow up an evaluation with a justification of the approach taken.
Information	INF	Providing new knowledge with respect to the nature of a subject.
Hypothesis	HYP	Expressing a personal representation of a subject. This representation is made through the use of expressions such as "I believe that…," "I think …," or "…maybe…"
Development	DEV	Presenting a new idea in detail. This is considered a creative activity.
Introduction	Intro	Introducing the subject.

Table 2. Types of activity found in review sequence moves

2.3 STEP C: Coder reliability

The coding scheme is applied on the protocol by coders. More than one coder is needed on a least part of the protocol to measure the inter-rater reliability. Coder A, who collaborated in defining the coding scheme, performed the first coding, called *Version A*. Coder B, trained by coder A, performed coding in many sessions, and is called *Version B*. This approach provides two coded versions of the same session with different coder characteristics. The codes were shown to be reliable, according to accepted practices (Cone, 1977; Cooil & Rust, 1994). Coding scheme must be validated such that inter-rater dissents are regarded as a source of variance, but not as a possible bias.

Table 3 presents the Perreault and Leigh (1989) reliability index and Cohen's Kappa (1960) for the two versions. These indices, obtained from two different statistical approaches, are above a threshold estimated to signify good agreement between coders.

Versions	Reliability Index	Std dev	Kappa
Version A and B	0.83	0.020	0.68

Table 3. Perreault and Leigh reliability index and Cohen's Kappa

2.4 STEP D: Model building

The last step is to build a model or a theory from the analysis of the coded moves. The purpose of this research was to identify typical configurations of exchanges (i.e. configurations occurring with a significant frequency) in review meetings. The model presented is based on the Latent Semantic Analysis (LSA) statistical methods (Sackett, 1979; Allison & Liker, 1982). The analysis of sequential structures using LSA is grounded in information theory. It enables the identification of units (moves, exchanges) that follow each other, with or without other units in between. The analysis consists of determining whether

or not the occurrence frequency of a given unit is independent of the occurrence frequency of another unit. These methods can be used in an iterative way, in several cycles, as long as LSA allows significant configurations to be detected.

This qualitative data analysis method is aimed at generating a model structure, and there are a number of approaches to strengthen, or confirm a proposition after it has been generated from the data. One method for helping to confirm findings, which is particularly well suited to most software engineering studies, is obtaining feedback on the findings from the subjects who provided the data in the first place. This strategy is sometimes called member checking (Seaman, 1999), and was the method used in this review meeting study. Participants, when presented with this model (see the box on the right-hand side of Figure 1), argue that it is only partially representative of the exchange patterns typical of a technical review meeting.

Confirmation of a qualitative model is a major endeavor. The whole qualitative approach is rigorous and objective, but the coding activity, which relies on the categorization of the protocol based on the coding scheme, may not be? What is the impact of coding variability on the derived model structure resulting from coder subjectivity? Would another coder deliver a different model structure?

Theoretically, each model could be different, depending on the intrinsic variability of sequences of coded moves. How can we know which model is the best? Assuming that it is feasible from a cost perspective to perform many codings and derive the corresponding model for each of them, it will be possible, for example, to consider only the model structure that is common to all models, or the features that are encountered most frequently.

This paper presents a novel approach based on Monte Carlo Coding Replication (MCCR), which enables the validation of the models by simulating numerous codings based on the inter-rater reliability measure. The approach consists of measuring the variability of the coding on a representative sample of the data, and then simulating that coding with the same variability for all the transcripts. The impact of the coding variability obtained from various simulated coders is then analyzed on the resulting model structures.

3. Simulation procedure

MCCR is an original simulation approach that has been developed to validate the model structure derived from protocol analysis. The main challenge is to characterize the variability of the coders in such a way as to be able to simulate sequences of coded moves that are within the inter-rater reliability level.

We present below a new approach to replicating coding activities based on Monte Carlo simulation. Underlying MCCR are the following two assumptions:

1. Reference sessions are representative of associated sessions, and consequently the variations between coders observed for one reference session are representative of the variations for any of the sessions associated with this reference session.
2. The simulated data will not generate hidden dependencies between the data, and the generated data will be similar to real data.

The first assumption is necessary in order to reduce the effort required to measure variability. In theory, all the sessions are unique and variability could be measured on each of them. Pragmatic considerations and the homogeneity of the sessions, in terms of reviewing activities, make this assumption reasonable. In case of doubt, or given enough resources, it could be validated on many sessions. The example used in this chapter to illustrate the MCCR is based on a single representative session. Many sessions were coded by two coders, and it was estimated by the coders that session 2 versions A and B were representative of all the sessions coded. The second assumption is guaranteed by the parameters of the normal distributions used for the simulations, which are specific to, and derived from, each session.

MCCR is based on two parameters. The first takes into account the qualitative and subjective differences observed between coders in the reference session to simulate the variability. The second takes into account the profile of moves within the real session to reconstruct the simulated sessions.

Figure 2 shows the three steps added (E to G) in the MCCR approach from the traditional approach. In Step E, the variability between coders, which is based on the reference coding sessions used to compute the inter-rater reliability index, is characterized. In Step F, these variability characteristics are fed to the simulator. In Step G, models are reconstructed from the simulated moves. Finally, a new model structure is derived based on the invariant structures, or the structures that have the greatest probability of occurrence.

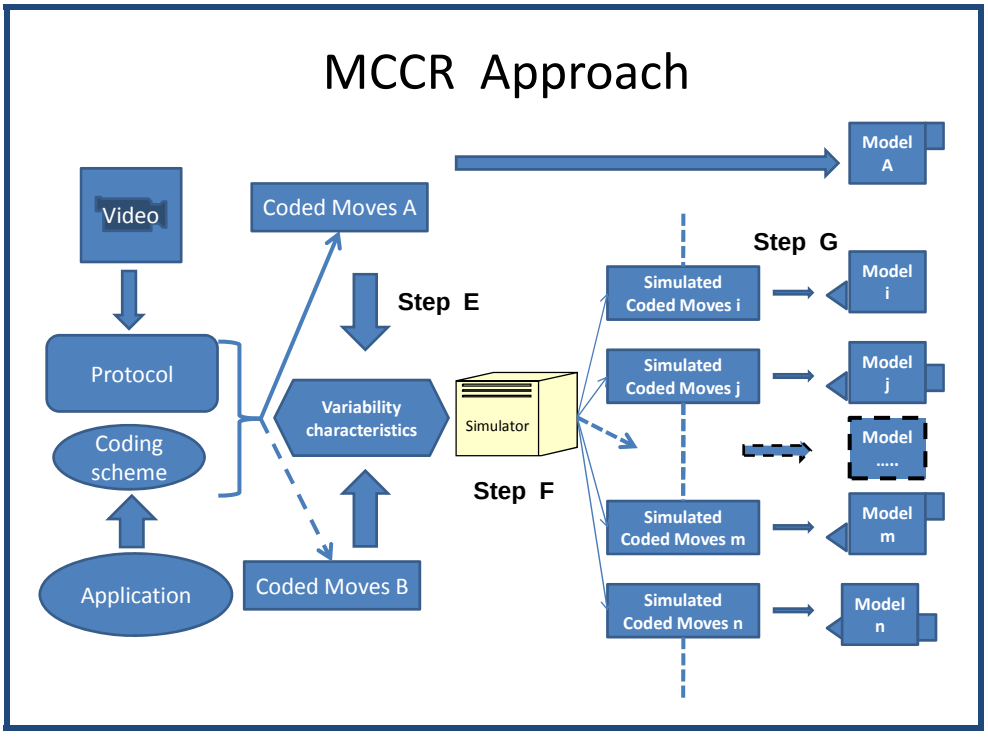

Fig. 2. Three added steps for the MCCR approach

3.1 Step E: Coded move variability characteristics

Coded move variability is characterized according to two dimensions: coders (E1) and moves (E2). Step E1 characterizes the variability of code occurrence originating from the coders. Step E2 characterizes the variability of move duration originating from the protocol. In this step, simulated codes are assigned to appropriate moves, since the session duration is the same regardless of the coders (human or simulated).

The reliability and appropriateness of the coding scheme constitute major issues and are the subject of much research from a variety of viewpoints (Carletta et al., 1997) (Craggs & Wood, 2005). The aim of the MCCR approach is to validate the resulting model structure by taking into account the variations in coder practices, assuming that the coding scheme is appropriate and the coders are reliable.

In **Step E1**, the variability for each code is characterized. For example, variation occurs when one coder evaluates a move as a DEV, while another considers it to be an EVA (See Table 2 for abbreviation meaning). The agreement matrix between two coders provides the probabilities of code occurrence.

Table 4 presents the number of occurrences of each code for Versions A and B for the reference session 2. For example, the element in the first line (DEV) and the first column (DEV) indicates that 11 moves were coded as DEV in both versions. The element in the second column (EVA) indicates that three moves coded DEV in Version A were coded EVA in Version B. Column JUS shows that no DEV move in Version A was coded JUS in Version B. The last column gives the total number of moves with a particular code in Version A. The last line gives the total number of moves with a particular code in Version B. For example, moves were coded DEV 20 times in Version A and 16 times in Version B, with 11 moves coded DEV in both versions. The values on the diagonal are the exact coded agreements for the two versions, which sums up to 106. An introductory move (INT), not shown in this table, is always the first move, the one that introduces the topic to be reviewed, and cannot be misjudged.

Version A	*Version B*					
	DEV	*EVA*	*JUS*	*HYP*	*INF*	*TOTAL*
DEV	**11**	3		2	4	20
EVA		**15**		2	4	21
JUS		2	**11**	1	5	19
HYP	5	1		**19**	3	28
INF			2	1	**50**	53
TOTAL	16	21	13	25	66	141

Table 4. Code occurrences from Version A and Version B of session 2

Table 3, presented previously, shows that the two coded versions A and B of the session 2 are reliable for discourse analysis. With the current state of the art, it is not possible to claim that one version is better than another. The main challenge is to understand and quantify the impact of this variability on the model structure derived from this protocol analysis, where only one coder coded all the sessions. It is very time-consuming, and almost

impossible, to code all seven sessions many times in order to measure the variability of the resulting codes, as shown in this case for session 2.

In **Step E2**, the move duration, which is an intrinsic property of moves within a session, is characterized. Table 5 and Table 6 show the average duration of a move and its standard deviation respectively for the moves of Versions A and B. For example, the 11 DEV moves, which were coded DEV in both versions, have an average duration of 21.3 seconds with a standard deviation of 11.8 seconds. The 3 DEV moves of Version A that were coded EVA in Version B have an average duration of 8 seconds, with a standard deviation of 1.7 seconds. These tables provide the duration characteristics of code variability.

Version A	*Version B*				
	DEV	*EVA*	*JUS*	*HYP*	*INF*
DEV	21.3	8		19	10.3
EVA		10.4		9.5	5
JUS		20	20.6	5	8.5
HYP	8	4		15.7	19.7
INF			9.5	3	15.1

Table 5. Average move duration in Versions A and B

Version A	*Version B*				
	DEV	*EVA*	*JUS*	*HYP*	*INF*
DEV	11.8	1.7		5.7	6.1
EVA		6.7		3.5	2.9
JUS		5	13.3	3	4.4
HYP	3.2	3		10,7	16.2
INF			4.9	2	12.9

Table 6. Standard deviation for move duration in Versions A and B

The three tables needed for Step E can be easily and automatically computed from the coded move sessions. This approach requires that all the moves in a session be defined before coding. All coders code the same number of moves in a given session. All the moves are identified in the protocol with a unique ID. For example, coders will not have to decide whether an utterance accounts for one or two moves.

3.2 Step F: Simulation procedure

The Monte Carlo simulation of a session uses the coder and move duration characteristics obtained in Step E to produce a simulated session. The simulation procedure is performed following five steps. Step F1 and F2 simulates a new sequence of codes according to the coder variability characteristics. Step F3, F4 and F5 associates each simulated code to a corresponding move, according to the duration characteristics of each type of code. The number of moves per session is not modified by the simulation. First, all simulated sessions have to be coded by one human coder, which is Version A (see Figure 2). The

reliability index and the Kappa coefficient are calculated for each simulated session based on Version A, and should be near or above the values computed in Table 3 to be considered reliable.

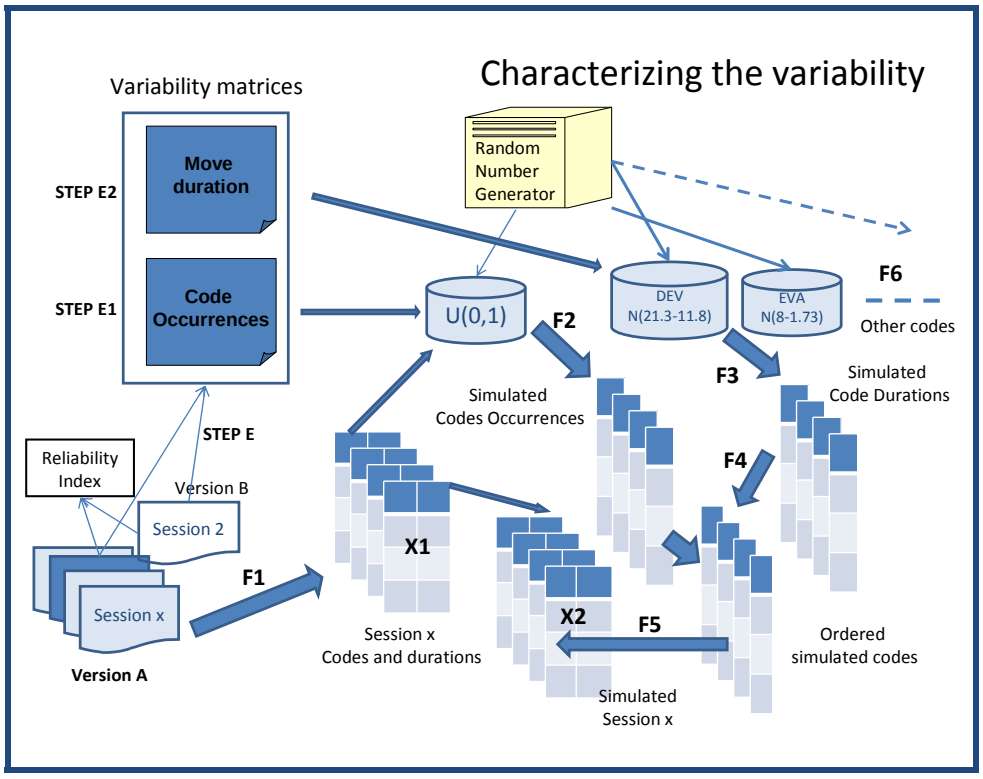

Fig. 3. Steps for the simulation of a session

Figure 3 illustrates the five steps of the simulation procedure, based on the simulation of session x. In the bottom-left corner is Version A of all the sessions that were hand-coded. Session 2 is one of the sessions of Version B that was selected by another coder as the reference session from which are derived the variability matrices produced in Step E1 and Step E2, described in the previous section. We recall that there can be more than one reference session. For example, sessions A1 to A4 of Version A can have session B2 of Version B as the reference session, and sessions A5 to A7 can have B5 as the reference session, etc.

The simulation procedure is composed of five steps, labelled F1 to F5 in Figure 3. The table dimensions shown in the figure are for illustration purposes only and do not correspond to the real example. The first step (F1) is to extract all the moves with the same type of activity, and their duration, from a given session x of Version A. For example, Table 7 (Table X1 in Fig. 3) presents the six DEV codes and their duration ordered according to move occurrence (ID). In the ID field, the letter representing the author is omitted, since it is irrelevant for this analysis.

ID	Code	Duration
4	DEV	15
12	DEV	4
13	DEV	37
25	DEV	12
32	DEV	20
46	DEV	8

Table 7. Duration of DEV codes to be simulated

The next step (F2) is to simulate the new codes that will correspond to the DEV codes assigned by the human coder. This new simulated code depends on the variability probability profile between coders.

The variability between coders is such that there is a probability that some of these DEV codes will be coded differently by another coder. This probability is obtained from the multi-coding performed on the reference session with different coders. The probabilities of code occurrence are presented in Table 8, which is derived from Table 4.

For example, the number in the first row and the first column (.55) represents 55% of the 20 total occurrences of DEV for Version A. The three occurrences of EVA account for 15% of the total occurrences. The probability of occurrence is expressed cumulatively, so EVA is at 70%.

Version A	***Version B***				
	DEV	***EVA***	***JUS***	***HYP***	***INF***
DEV	0.55	0.7	0.7	0.8	1
EVA	0	0.71	0.71	0.81	1
JUS	0	0.1	0.69	0.74	1
HYP	0.18	0.21	0.21	0.89	1
INF	0	0	0.04	0.06	1

Table 8. Probability of occurrence of codes

Six random numbers generated from a uniform distribution U (0, 1) are used to determine how many DEV codes, if any, will be coded differently. The ordered random numbers in the first column of Table 9 are compared with the probabilities on the DEV line of Table 8. In Table 9, there are four numbers lower than 0.55 (in bold) and two larger than 0.55, but less than 0.7, which means that the simulated session will have four DEV codes and two EVA codes, as shown in the second column of Table 9.

Ordered Random number	Simulated Assigned Code
0.13	DEV
0.22	DEV
0.40	DEV
0.49	DEV
0.57	EVA
0.68	EVA

Table 9. Simulated code occurrences

The next step (F3) is to determine which of the two DEV codes in the real session will be interpreted as an EVA code in the simulated session. Ordering and sequencing the codes is important, since many analyses like LSA rely on the code sequences. The assignment of the simulated codes to corresponding moves is based on the duration patterns of the moves, as characterized in Table 5 and Table 6. This information is used to define the characteristics of the virtual duration of each of the new DEV and EVA codes. It is called virtual duration, because it is used only to find the appropriate moves, since the real duration of a move is fixed, regardless of how it is coded (human or simulated). The simulation is based on normal distributions with the parameters given in Table 5 and Table 6 for the corresponding codes. Normal distributions N(21.3-11.8) and N(8-1.73), which have been found to be appropriate, are used for DEV and EVA respectively. Analyses of the moves are performed to determine the most appropriate distributions for code duration simulations. Table 10 presents the results for the four DEV codes (in bold) and the two EVA codes in increasing order of simulated duration.

The next step (F4) is to match the simulated codes with the real moves. Table 11 shows the resulting association of simulated codes with the real codes. The shorter simulated code as given by Table 10 is associated with the shorter real code, and so on. The simulated duration is only used to find the code association.

CODE	SIMULATED DURATION
EVA	7
DEV	**9**
EVA	10
DEV	**17**
DEV	**25**
DEV	**28**

Table 10. Simulated code durations

ID	*CODE*	*Duration*	*Sim CODE*	*Virtual Duration*
12	DEV	4	EVA	7
46	DEV	8	**DEV**	**9**
25	DEV	12	EVA	10
4	DEV	15	**DEV**	**17**
32	DEV	20	**DEV**	**25**
13	DEV	37	**DEV**	**28**

Table 11. Matching simulated codes on the basis of duration ordering

The last step (F5) is to build a new simulated version according to the simulated codes assigned to the real moves. Table 12 shows the ordered DEV codes, according to their ID, in the real and simulated versions of session x. The process is performed for all the codes of Version A to build a full, simulated version that has the same duration and where qualitative variations of coding are performed according to the coder's characteristics provided by the reference session.

MCCR provides a new version of the human-coded version that is likely to have the same characteristics as a version obtained by another coder. It will have the same number of codes with a variability that will be within the characteristics measured in the reference session. For example, Table 12 shows that a simulated coder will have coded moves 4, 13, 32 and 46 with the same code DEV as the human coder but will have coded moves 12 and 25, which are lasting 4 and 12 second respectively, as EVA codes. Kappa index shows that such coding is within the inter-reliability level found between the two human coders.

ID	Human-coded Version	Simulated Version	Real Duration
4	DEV	**DEV**	15
12	DEV	EVA	4
13	DEV	**DEV**	37
25	DEV	EVA	12
32	DEV	**DEV**	20
46	DEV	**DEV**	8

Table 12. DEV code sequence in coded and simulated versions

The whole process of simulating coded sessions is summarized by the two matrices, labelled X1 and X2 at the bottom center of Figure 3. The left-most column containing the coded type of activity (DEV, EVA, etc.) is used for the simulation process, since this is the source of the variability, while the right-most column containing the move duration is transferred 'as is' (arrow with no label) in the resulting simulated sessions (X2). The same simulation process (F6) is repeated for each type of activity, as shown in Table 8.

MCCR is efficient and could be performed hundreds of times in any given session, which means that the whole spectrum of coding variability can be explored. The Kappa coefficient is used to evaluate the level of agreement between the simulated versions and the real version.

4. Reconstructing the models (Step D)

Reconstruction of the models (Figure 1, Step D) is the last step of protocol analysis. This section shows the impact of the MCCR approach on validating the exchange pattern model structure derived from protocol analysis performed by a single coder and on the model derived from multi-simulated coding.

The question MCCR has answered is: How do we know that the model structure would have been the same if another coder had performed the protocol analysis?

The analysis of model structure variability is not part of the MCCR approach. It is presented only to illustrate the usefulness of MCCR in a real case. Readers interested in the detail of model building and measurement of the link significance levels for the model structures are referred to the papers presenting the models (D'Astous et al, 2002, 2004).

The purpose of this protocol analysis was to build a model of the communication exchanges occurring in technical review meetings, in order to optimize their efficiency and to better integrate these activities into the software engineering process.

Figure 4 shows the model structure resulting from the single-coder model on the left-hand side and model structure from the simulated model on the right-hand side. The single-coder model is based on the analysis of seven coded meetings. The patterns observed in each meeting are added together to create the left-hand cumulative pattern of the figure. The right-hand cumulative pattern is made-up of a combination of the simulated meetings, where each of the seven meetings is simulated 10 times. More simulation did not add any new information to the model structure. LSA is applied to each meeting independently.

The following briefly explains the model. The exchanges identified (Table 2) introduce the subject (INTRO), provide information (INF), make a hypothesis (HYP), generate a development (DEV), evaluate (EVAL), and justify (JUS) viewpoints. Each meeting is analyzed and the number of link occurrences is indicated by the size of the link. Links in large bold type indicate that they are statistically significant for all meetings, while solid narrow type indicates that the links are statistically significant for some of the meetings.

The single-coder model provides the following exchange patterns during a peer review meeting. Any sequence of exchanges starts with the introduction (INTRO) of the topic to be reviewed. Following the INTRO, there are three significant exchanges that could occur during a session: a development exchange DEV, an evaluation exchange EVA, or an information exchange INF. Peers are likely to continue on into DEV and EVA exchange sequences or into INF and HYP exchange sequences. Sometimes, a justification exchange JUS will follow an evaluation EVA or a hypothesis HYP exchange, and a justification JUS exchange may be followed by an information INF exchange. The single-coder model also indicates that an introduction INTRO may be followed by a hypothesis HYP exchange.

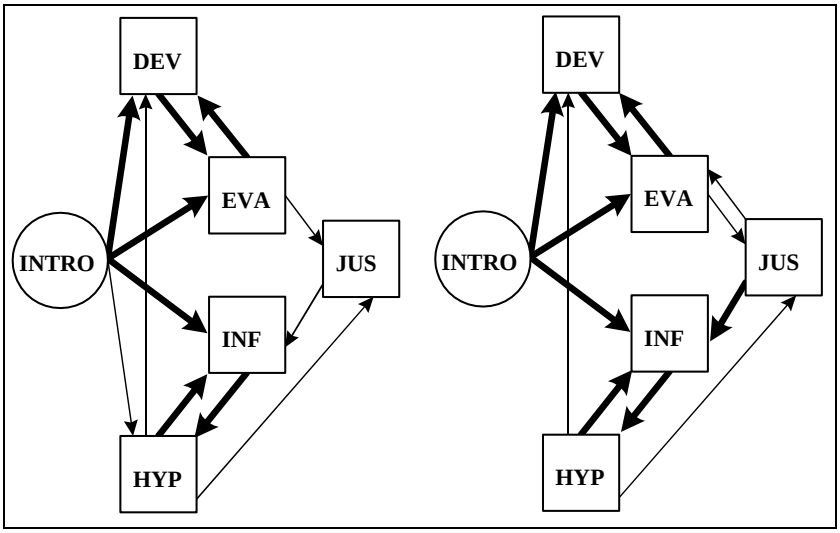

Fig. 4. Single-coder model (left-hand side) and simulated model (right-hand side).

At first glance, the single-coder and simulated patterns are similar, which means that the variability in coding is not dramatic. Nevertheless, there are some differences that emerge from the simulated model. For example, all but one of the large bold links is common to

both models, except for the JUS-INF link, which appears only in the simulated model on the right-hand side of the figure. Secondary links, represented by solid lines, are also almost the same in the two models, except for the INTRO-HYP links, which appear only in the single-coder model on the left-hand side of the figure, and the JUST-EVA link, which appears only in the simulated model.

The simulated model confirmed the salient features of the single-coder model. MCCR may resolve ambiguities that arise from the qualitative components of coding. For example, the exchange patterns INTRO and HYP observed in the single-coder model are clearly not significant based on the simulated model. In contrast, the weak link between JUS and INF observed in the single-coder model is, in fact, a significant one in the simulated model. The simulated model shows the occurrence of an evaluation-justification loop.

5. Conclusion

MCCR enables analysis of the impact of individual qualitative variability on coding transcripts from protocol analysis. It is much easier to perform than real replication of the protocol analysis by different coders, and requires far fewer resources. Moreover, it provides information on the components of the resulting models, which are sensitive to the qualitative components of coding. This Monte Carlo validation approach is generic, and not limited or specific to the case study presented. The simulation is essentially based on the number of categories, which is limited by the capacity of human beings to handle numerous categories. This number is always very small compared to the power of computer simulation.

To summarize, MCCR is performed on complete protocol analysis performed by a well-trained coder. The coding is redone in one or many representative sessions, called reference sessions, by another well-trained coder. The reliability of the two coded protocols is measured with known tools, such as agreement indices or Kappa coefficients. Qualitative differences between the coders, which are interpreted as the qualitative variations in the coding pattern, are characterized by a probability matrix derived from the two codings of the reference sessions. Profiles of coded moves are characterized by the average and standard deviation of their durations. These parameters are used to specify the parameters of the normal distributions of the random generator. All codes are simulated according to the distribution specifications provided by the reference sessions. Sequences of simulated codes are built according to the specific code duration within a given session. A simulated session of exactly the same duration is obtained with a coding variability that is within the range shown for the reference session. The Kappa coefficient is computed on the simulated sessions to validate their reliability. As many simulations as needed could be performed to obtain a statistically significant and stable model.

Observational studies are difficult to conduct in an industrial environment, and analysis of the data is very time-consuming. However, such studies constitute one of the few approaches that enable the modeling of real activities. MCCR is a new method developed to extract from the observed model structure the components related to the subjective and qualitative components of the observations. This method has been illustrated with an example based on modeling the technical review meeting, but it is of general interest and can be applied to any observational study.

6. Acknowledgment

This research would not have been possible without the generous participation and patience of the software development team members from whom the data were collected. To all these people, we extend our grateful thanks. This research was supported in part by NSERC grant A-0141.

7. References

Allison, P.D. & Liker, J.K. (1982). Analyzing sequential categorical data on dyadic interaction: a comment on Gottman, *Psychology Bulletin*, 91, 393-403.

Baer, D.M. (1977). Reviewer's Comment: Just because it's reliable doesn't mean that youcan use it, *Journal of Applied Behavior Analysis*, 10 (1), 117-119.

Barnas Y. (1996). Formal aspects of model validity and validation in system dynamics, *System Dynamics Review*, 12 (3), 183-211.

Berk, R.A. (1979). Generalizability of behavioral observations: A clarification of interobserver agreement and interobserver reliability. *American Journal of Mental Deficiency*, 83 (5), 460-472.

Campbell, D.T. & Fiske, D. (1959), Convergent and discriminant validation by the multi-trait, multi-method matrix. *Psychological Bulletin*, 56, 81-105.

Carletta, J., Isard, S., Doherty-Sneddon, G., Isard, A., Kowtko, J.C., & Anderson, A.H. (1997). The reliability of a dialogue structure coding scheme, *Computational Linguistics*, 23(1), 13-31.

Cicchetti, V.D., Shoinralter, D., & Tyrer, P.J. (1985). The effect of number of rating scale categories on levels of interrater reliability: A Monte Carlo investigation, *Applied Psychological Measurement*, 9 (1), 31-36.

Cohen, J. (1960). A coefficient of agreement for nominal scales, *Educational and Psychological Measurement*, 20(1), 37-46.

Cone, J.D. (1977). The relevance of reliability and validity for behavioral assessment, *Behavior Therapy*, 8, 411-426.

Cooil, B. & Rust, R.T. (1994). Reliability and expected loss: A unifying principle. *Psychometrika,* 59(2), 203-216.

Craggs, R., & Mc Gee Wood, M. (2005). Evaluating discourse and dialogue coding schemes, *Computational Linguistics*, 31(3), 289–296.

Cronbach, L.J., Gleser, G.C., Nanda, H., & Rajaratnam, N. (1972). *The dependability of behavioral measurements: Theory of generalizability for scores and profiles*. John Wiley & Sons, New York.

Curtis, B. (1980). Measurement and experimentation in software engineering. *Proceedings of the IEEE*, 68(9), 1144-1157.

D'Astous, P., & Robillard, P.N. (2002). Empirical study of exchange patterns during software peer review meetings, *Information and Software Technology*, 44, 639-648.

D'Astous, P., Détienne F., Visser W., & Robillard, P.N. (2004). Changing our view on design evaluation meetings methodology: A study of software technical review meetings, *Design Studies*, 25(6), 625-655.

Edmonson, W. (1981). *Spoken discourse: A model for analysis*. London: Longman,.

Ericsson, K.A., & Simon, H.A. (1993). *Protocol analysis: Verbal reports as data*, Revised Edition. MIT Press, Cambridge.

Foster, A., Urquhart, C. & Turner, J. (2008). "Validating coding for a theoretical model of information behaviour". *Information Research*, 13(4) paper 358.

Herbert, J., & Attridge, C. (1975). A guide for developers and users of observation systems and manuals. American Educational Research Journal, 12(1), 1-20.

Hughes, J., & Parkes, S. (2003). Trends in the use of verbal protocol analysis in software engineering research, *Behaviour & Information Technology*, 22(2), 127-140.

Johnson, S. M., & Bolstad, O.D. (1973). Methodological issues in naturalistic observation: Some problems and solutions for field research. In Hamerlynck, L. A., Handy, L. C., Mash, E. J. (eds.) *Behavior Change: Methodology, Concepts and Practice.* Research Press, Champaign, Illinois, 7-67.

Khaled, E.E. (1999). Benchmarking Kappa: Interrater agreement in software process assessments, Empirical Software Engineering, 4, 113-133.

Kerlinger, F.N. (1973), *Foundations of behavioralrResearch*, 2nd edition. Holt, Rinehart and Winston, NY,

Mayrhauser, von A., & Lang, S. (1999). A coding scheme to support systematic analysis of software comprehension, *IEEE Transaction on Software Engineering*, 25(4), 526-540.

Medley, D.M. & Mitzel, H.E. (1963). *Measuring classroom behavior by systematic observation*. In Gage, N. L. (ed.) Handbook of Research on Teaching. Rand McNally, Chicago, 247-328.

Mitchell, S.K. (1979). Interobserver Agreement, Reliability, and Generalizability of Data Collected in Observational Studies. *Psychological Bulletin*, 86(2), 376-390.

Nelson, R.O., Hay, L.R., & Hay, W.M. (1977), Comments on cone's "The relevance of reliability and validity for behavioral assessment". *Behavior Therapy*, 8, 427-430.

Olson, G.M., Herbsleb, J.D., & Rueter, H.H. (1994), Characterizing the sequential structure of interactive behaviors through statistical and grammatical techniques. *Human-Computer Interaction*, 9, 427-472.

Perreault, W.D.J., & Leigh, L.E. (1989), Reliability of nominal data based on qualitative judgments, *Journal of Marketing Research*, 26, 135-148.

Robillard, P.N., d'Astous, P., Détienne, F., & Visser, W. (1998), Measuring cognitive activities in software Engineering. *Proceedings of the 20th International Conference on Software Engineering*, 292-300.

Sackett, G.P. (1979), The lag sequential analysis of contingency and cyclicity in behavioral interaction research. In Osofsky, J. D. (ed.) *Handbook of Infant Development*. Wiley-Interscience Publication, 623-649.

Sanderson, P.M.,& Fisher, C. (1994). Exploratory sequential data analysis: Foundations. *Human-Computer Interaction*, 9, 251-317.

Seaman, C.B. (1999), Qualitative methods in empirical sudies of software engineering, *IEEE Transaction on Software Engineering*, 25(4), 557-572.

Sinclair, A., & Coulthard, R.M. (1975). Towards an analysis of discourse. *The English used by Teachers and Pupils*. Oxford University Press.

Trickett, S., & Trafton, J.G. (2002), A primer on verbal protocol analysis, in *Handbook of Virtual Environments: Design, Implementation, and Applications (Human Factors and Ergonomics),* Kay Stanney (Editor), 332-346.

Weber, R.P. (1990). Basic content analysis, Series: *Quantitative Applications in the Social Sciences*, Sage University Paper