The Journal of Systems & Software 202 (2023) 111707

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Reliability in software engineering qualitative research through Inter-Coder Agreement☆

Ángel González-Prieto [a,b,*], Jorge Perez [c], Jessica Diaz [c], Daniel López-Fernández [c]

[a] Facultad de Ciencias Matemáticas, Universidad Complutense de Madrid, Plaza Ciencias 3, Madrid, 28040, Spain
[b] Instituto de Ciencias Matemáticas (CSIC-UAM-UCM-UC3M), C/ Nicolás Cabrera 13-15, Madrid, 28049, Spain
[c] ETSI de Sistemas Informáticos, Universidad Politécnica de Madrid, C/ Alan Turing s/n, Madrid, 28031, Spain

ARTICLE INFO

Article history:
Received 21 July 2022
Received in revised form 10 April 2023
Accepted 12 April 2023
Available online 19 April 2023

Dedicated to the memory of Klaus Krippendorff

Keywords:
Inter-Coder Agreement
Krippendorff's αcoefficient
Coding
Qualitative research
Software engineering

ABSTRACT

The research on empirical software engineering that uses qualitative data analysis is increasing. However, most of them do not deepen into the validity of the findings, specifically in the reliability of coding in which these methodologies rely on. This paper aims to establish a novel theoretical framework that enables a methodological approach for conducting this validity analysis through Inter-Coder Agreement (ICA), based on the use of coefficients to measure the degree of agreement in collaborative coding. We systematically review several existing variants of Krippendorff's α coefficients and provide a novel common mathematical framework to unify them. Finally, this paper illustrates the use of this theoretical framework in a large case study on DevOps culture. We expect that this work will help researchers who are committed to measuring consensus with quantitative techniques in collaborative coding, conducted as part of a qualitative research, to improve the rigor of their findings.

1. Introduction

In recent years, empirical software engineering that uses qualitative research methods is on the rise (Ghanbari et al., 2018; Stol et al., 2016; Wohlin et al., 2012; Salleh et al., 2018; Storer, 2017). Indeed, today many studies can be found in the area of software engineering that use qualitative methods (Bhasin et al., 2021; Pereira et al., 2022; Madampe et al., 2023; Berntzen et al., 2023), and adaptations from well-known research methods to socio-technical contexts are being performed such as in the case of grounded theory (Hoda, 2022). In this regard, coding plays a key role in the qualitative data analysis process of case studies, interview surveys, and grounded theory studies. Content analysis, thematic analysis, and grounded theory coding methods (for example, in vivo, process, initial, focused, axial, and theoretical coding) have been established as top-notch procedures to conduct qualitative data analysis as they provide methods for examining and interpreting qualitative data to understand what they represent (Cruzes and Dyba, 2011; Saldaña, 2012).

Reliability in coding is particularly crucial to identify mistakes before the codes are used in developing and testing a theory or model. In this way, it assesses the soundness and correctness of the drawn conclusions, with a view towards creating well-posed and long-lasting knowledge. Weak confidence in the data only leads to uncertainty in the subsequent analysis and generates doubts about findings and conclusions. In Krippendorff's own words: "If the results of reliability testing are compelling, researchers may proceed with the analysis of their data. If not, doubts prevail as to what these data mean, and their analysis is hard to justify" (Krippendorff, 2018).

This problem can be addressed by means of well-established statistical techniques known as Inter-Coder Agreement (ICA) analysis. These are a collection of coefficients that measure the extent of agreement/disagreement between several judges when they subjectively interpret a common reality (Cohen, 1960; Fleiss, 1971; Krippendorff, 2018; Scott, 1955). In this way, these coefficients allow researchers to establish a value of reliability of the coding that will be analyzed later to infer relations and lead to conclusions. Coding is reliable if the coders can be shown to agree on the categories assigned to the units to an extent determined by the purposes of the study (Craggs and Wood, 2005).

ICA techniques are used in many different research contexts from both social sciences and engineering, and recently socio-technical research (Hoda, 2022). For example, these methods have been applied to assess (i) the selection criteria of primary studies and data extraction in systematic literature reviews (Pérez

☆ Editor: Burak Turhan.

* Corresponding author at: Facultad de Ciencias Matemáticas, Universidad Complutense de Madrid, Plaza Ciencias 3, Madrid, 28040, Spain.
E-mail addresses: angelgonzalezprieto@ucm.es (Á. González-Prieto), jorgeenrique.perez@upm.es (J. Perez), yesica.diaz@upm.es (J. Diaz), daniel.lopez@upm.es (D. López-Fernández).

https://doi.org/10.1016/j.jss.2023.111707

et al., 2020) as well as (ii) the ratings when coding qualitative data (Díaz et al., 2023). This has led to a variety of related terms interchangeably used, such as inter-coder, inter-rater, or inter-judge agreement, according to the custom of the topic.

However, although more and more researchers apply inter-rater agreement to assess the validity of their results in systematic literature reviews and mapping studies as we analyzed in Pérez et al. (2020), it is a fact that few studies in software engineering analyze and test the reliability and trustworthiness of coding, and thus the validity of their findings. Some of the authors of this research were involved in conducting a Systematic Mapping Study (SMS) aimed at testing whether the use of ICA techniques in Grounded Theory studies in the field of software engineering is a common practice (Díaz et al., 2023). In most of the 49 primary studies analyzed, Krippendorff's α or Cohen's κ coefficients were used, but in a very shallow way with a mere mention to the final value of the statistics without any further analysis of the underlying causes.

Similar results were obtained in a systematic literature review reported by Nili et al. (2020) in information management research. In this later study, the authors analyzed the use of ICA for qualitative research in works published in three journals during the period 2013–2019. They concluded that 29 papers applied Cohen's κ coefficient, 26 used a simple Percent Agreement, and 6 used Krippendoff's α coefficient. As it is well known, Cohen's κ coefficient is not a suitable indicator since it presents two anomalies due to the prevalences in the marginal totals, whereas Percent Agreement does not take into account the random agreements in its computation (see Sections 2.1 and 2.3). Moreover, the studies that applied Krippendorff's α coefficient only showed the final result of this metric, without any kind of mention to the computation procedure, context conditions or interpretation. In all these works, a simple mention to the "Krippendorff's α coefficient" is provided, without any distinction between the different variants of this statistic. However, as we will argue in the upcoming sections, the interplay between these variants and their discrepancies in the results are crucial to rigorously assess the agreement between coders and effectively detect disagreements.

In this paper, we propose to introduce the ICA analysis techniques in software engineering empirical research to enhance the reliability of coding in qualitative data analysis and the soundness of the results. For that purpose, in Section 2 we review some of the coefficients that, historically, have been reported in the literature to measure the ICA. We start discussing, in Section 2, some general purpose statistics, like Cronbach's α (Cronbach, 1951), Pearson's r (Pearson, 1895) and Spearman's ρ (Spearman, 1904), that are typically misunderstood to be suitable for ICA analysis, since they cannot measure the degree of agreement among coders but correlation, a weaker concept than agreement. Section 2.1 reviews some coefficients that evaluate the agreement between coders, which are not suitable for measuring reliability because they do not take into account the agreement by chance, such as the percent agreement (Feng, 2014, 2015) and the Holsti Index (Holsti, 1969). We also present a third group, with representants like Scott's π (Scott, 1955) (Section 2.2), Cohen's κ (Cohen, 1960) (Section 2.3) and Fleiss' κ (Fleiss, 1971) (Section 2.4), which have been intensively used in the literature for measuring reliability, especially in social sciences. However, as pointed out in Hayes and Krippendorff (2007), all of these coefficients suffer some kind of weakness that makes them non-optimal for measuring reliability. Finally, in Section 2.5 we sketch briefly Krippendorff's proposal to overcome these flaws, the so-called Krippendorff's α coefficient (Krippendorff, 2018), on which we will focus throughout this paper.

Despite the success and widespreadity of Krippendorff's α, there exists in the literature plenty of variants of this coefficient formulated quite ad hoc for very precise and particular situations, like Krippendorff (1970, 1995), Yang and Green (2011), Gwet (2011), De Swert (2012), Krippendorff et al. (2016) and Krippendorff (2018). The lack of uniform treatment of these measures makes their use confusing, and the co-existence of different formulations and diffuse interpretations become their comparison a hard task. To address this problem, Sections 3 and 4 describe a novel theoretical framework that reduces the existing variants to a unique universal α coefficient by means of labels that play the role of meta-codes. With this idea in mind, we focus on four of the most outstanding and widely used α coefficients and show how their computation can be reduced to the universal α by means of a simple relabeling. This framework provides new and more precise interpretations of these coefficients that will help detect flaws in the coding process and correct them easily on the fly. Moreover, this interpretation in terms of labels sheds light on some awkward behaviors of the α coefficients that are very hard to understand otherwise.

Section 5 includes a methodological proposal on the use and interpretation of Krippendorff's α coefficients to provide reliability in software engineering case studies. We illustrate this methodology with a running example based on a real case study extracted from the literature on a qualitative inquiry in the DevOps domain (Díaz et al., 2021).

Lately, in Section 6 we discuss the main limitations and threats to the validity of the proposed approach. Finally, in Section 7 we summarize the main conclusions of this paper and provide some guidelines on how to apply this tutorial to case studies in qualitative research. We expect that the theoretical framework, the methodology, and the subsequent guide introduced in this paper may help empirical researchers, particularly in software engineering, improve the quality and soundness of their studies.

Notice that, in this paper, we focus on testing reliability of qualitative data analysis through a very fine analysis of the underlying combinatorial framework of Krippendorff's α coefficients, specifically addressing reliability of coding. In this sense, this work is radically different from other contributions existing in the literature such as Nili et al. (2017a, 2020) and Nili et al. (2017b). The goal of these previous work is to assess the reliability in qualitative research by improving data collection, data processing, and subsequent analysis. In sharp contrast, our work addresses a much more fundamental issue: the fine analysis of disagreements during the coding. In other words, whereas Nili et al. use ICA measures as a black-box indicator of the quality of the coding and focus on the surrounding process, ICA is a central white-box tool in our study. In particular, the combinatorial framework developed in this work to reinterpret Krippendorff's α coefficients allows us to carefully track the origin of coding disagreements, find the radical problem in the codebook definitions, and thus it leads to a methodology that generates huge improvements in the ICA per coding round.

Finally, it is worth mentioning that, in some domains, such as social sciences, the use of ICA is widely accepted. In contrast, in the field of computer science, many authors reject the use of quantitative techniques in qualitative research. We are fully aware that qualitative research arises from a relativistic ontological stance, and thus aspects such as validity or reliability have a different fitting here than in quantitative research. However, it is not uncommon to apply quantitative instruments to qualitative research as a possible way to confer validity and methodological strength to the findings discovered within the qualitative paradigm. We do interpret the notion of "reliability" as a attempt to establish a single reality, but rather as an approach to develop a shared understanding that can also establish consistency among coders. As this paper shows, these quantitative approaches are not confronted with qualitative ones but can also complement

them. We expect that this work will help researchers who are committed to measuring consensus on collaborative coding, independently of the reasons why they decided to conduct this collaborative coding, their epistemological stance, or their belief that collaborative coding improves rigor.

1.1. Contribution of the work

The main achievements of the present work can be summarized as follows:

1. We introduce a novel universal formulation for Krippendorff's α coefficient as a cornerstone ICA metric. This conceptualization is presented through the formalism of set theory. To the best of our knowledge, this is the first rigorous formulation of these coefficients.
2. We show that this universal formulation gives rise to a new framework in which the existing variants of Krippendorff's α coefficient can be naturally interpreted. In this context, the variety of instances of the α coefficients existing in the literature naturally shows up as mere examples of a single statistical coefficient after a relabeling process.
3. We provide novel methods to detect subtle sources of disagreement during the coding process. This can only be achieved through the reinterpretation of the variants of the α coefficients as a relabeling process introduced in this work. This allows us to trace back the origins of disagreements directly to the source matter and to detect the semantic defects of the codebook that lead to these disagreements.
4. We propose a methodological approach, based on the novel interpretation of Krippendorff's α coefficients, to boost the use of ICA techniques in qualitative studies in Software Engineering. This methodology seeks to improve the reliability and soundness of these empirical studies through statistical methods. The approach is illustrated with a running example about DevOps adoption in companies.
5. We show that it is possible to integrate quantitative research techniques into qualitative research, regardless of the epistemological/ontological stance adopted by the latter.

2. Background

In a variety of situations, researchers have to deal with the problem of judging some data. Although the observed data are objective, the perception of each researcher is deeply subjective. In many cases, the solution is to introduce several judges, typically referred to as coders, to reduce the amount of subjectivity by comparing their judgments. However, in this context, a method for measuring the degree of agreement that the coders achieve in their evaluations, that is, the coding of the raw data, is required. Therefore, researchers must measure the reliability of the coding. Only after establishing that the reliability is sufficiently high, it makes sense to proceed with the analysis of the data (Krippendorff, 2018).

It is worth mentioning that, although often used interchangeably, there is a technical distinction between the terms agreement and reliability. Inter-Coder Agreement (ICA) coefficients assess the extent to which the responses of two or more independent raters are concordant; on the other hand, Inter-Coder Reliability (ICR) evaluates the extent to which these raters consistently distinguish between different responses (Gisev et al., 2013). In other words, the measurable quantity is the ICA, and using this value, we can infer reliability. In the same vein, we should not confuse reliability and validity. Reliability deals with the extent to which the research is deterministic and independent of the coders, and in this sense it is strongly tied to reproducibility; whereas validity deals with the truthfulness, with how the claims assert the truth. Reliability is a must for validity, but does not guarantee it. Several coders may share a common interpretation of reality, so that we have a high level of reliability, but this interpretation might be wrong and biased, so the validity is really small (ATLAS.ti Scientific Software Development GmbH, 2020).

Several statistical coefficients that have been applied for evaluating ICA exist in the literature, such as Cronbach's α, Pearson's r and Spearman's ρ. However, these coefficients cannot be confused with methods of inter-coder agreement test, as none of these three types of methods measures the degree of agreement among coders. For instance, Cronbach's α (Cronbach, 1951) is a statistic for interval or ratio level data that focuses on the consistency of coders when numerical judgments are required for a set of units. As stated in Nili et al. (2017a): "It calculates the consistency by which people judge units without any aim to consider how much they agree on the units in their judgments", and as written in Hayes and Krippendorff (2007) "[it is] unsuitable to assess reliability of judgments".

On the other hand, correlation coefficients, such as Pearson's r (Pearson, 1895) or Spearman's rank ρ (Spearman, 1904), measure the extent to which two logically separate interval variables, say X and Y, covary in a linear relationship of the form $Y = a + bX$. They indicate the degree to which the values of one variable predict the values of the other. On the contrary, the agreement coefficients must measure to what extent $Y = X$. High correlation means that the data approximate some regression line, whereas high agreement means that they approximate the 45 degree line (Krippendorff, 2018).

For these reasons, more advanced coefficients for measuring agreement are needed to assess reliability. In this section, we analyze several proposals that have been reported in the literature for quantifying the Inter-Coder Agreement (ICA) and inferring, from this value, the reliability of the coding.

2.1. Percent agreement

This measure is computed as the rate between the number of times the coders agreed when classifying an item (a datum to be analyzed) with the total amount of items (multiplied by 100 if we want to express it as a percentage). It has been widely used in the literature due to its simplicity and straightforward calculation (Feng, 2014, 2015). However, it is not a valid measure for inferring reliability in case studies that require a high degree of accuracy, since it does not take into account the agreement by chance (Nili et al., 2017a) and, according to Hayes and Krippendorff (2007), there is no clear interpretation for values different from 100%. In addition, it can be used only by two coders and only for nominal data (Zhao et al., 2013).

Table 1 shows an illustrative example, extracted from the Systematic Literature Review (SLR) (Pérez et al., 2020). This table shows the selection of primary studies; each item corresponds to one of these studies and each coder (denoted by J_1 and J_2) determines, according to a pre-established criterion, whether the studies should be promoted to an analysis phase (Y) or not (N). From these data, the percent agreement attained is $(10/15) \cdot 100 = 66.7\%$. At first sight, this seems to be a high value that would lead to a high reliability of the data. However, we miss the fact that the coders may achieve agreement purely by chance. As pointed out by Krippendorff (2018): "Percent-agreement is often used to talk about the reliability between two observers, but it has no valid reliability interpretations, not even when it measures 100%. Without reference to the variance in the data and to chance, percent agreement is simply uninterpretable as a measure of

Table 1
Decision table of two coders in a SLR.

Item:	#01	#02	#03	#04	#05	#06	#07	#08
J_1	N	N	N	N	N	N	Y	N
J_2	Y	Y	N	N	N	Y	Y	Y
Item:	#09	#10	#11	#12	#13	#14	#15	
J_1	N	Y	Y	N	N	N	Y	
J_2	N	Y	Y	N	Y	N	Y	

Table 2
Expected percent agreement for the data of Table 1.

	J_1	J_2	p_i	p_i^2
Y	4	9	4/30 + 9/30 = 0.433	0.188
N	11	6	11/30 + 6/30 = 0.567	0.321
Total				0.509

reliability—regardless of its popularity in the literature". Indeed, the following sections show the values of other ICA measures such as Cohen's κ and Krippendorff's α, which are very low: 0.39 (39%) and 0.34 (34%), respectively.

It is worth mentioning that there exists a variation of the simple percent agreement called the Holsti index (Holsti, 1969). It allows researchers to address the case in which the matter to be analyzed is not pre-divided into items to be judged, so that each coder selects the matter that considers relevant. However, for the same reasons as the simple percent agreement, this index is not a valid measure to analyze the ICA.

2.2. Scott's π

This index, introduced in Scott (1955), is an agreement coefficient for nominal data and two coders. The method corrects the percent agreement by taking into account the agreement that can occur between the coders by chance. The index of Inter-Coder Agreement for Scott's π coefficient is computed as

$$\pi = \frac{P_o - P_e}{1 - P_e}.$$

Here, P_o (observed percent agreement) represents the percentage of judgments on which the two analysts agree when coding the same data independently; and P_e is the percent agreement to be expected on the basis of chance. This latter value can be computed as

$$P_e = \sum_{i=1}^{k} p_i^2,$$

where k is the total number of categories and p_i is the proportion of the entire sample that falls in the ith category.

For instance, for our example of Table 1, we have $P_o = 0.667$, as computed in Section 2.1. For P_e, it is given by $P_e = (13/30)^2 + (17/30)^2 = 0.509$. Observe that, while the number of items is 15, in the previous computation we divided by 30. This is due to the fact that there are 15 items, but 30 *pairs of evaluations*, see also Table 2.

Therefore, Scott's π coefficient has a value of

$$\pi = \frac{0.667 - 0.509}{1 - 0.509} = 0.322.$$

2.3. Cohen's κ

Cohen's κ coefficient measures the concordance between two judges' classifications of m elements into k mutually exclusive categories. Cohen defined the coefficient as "the proportion of chance-expected disagreements which do not occur or, alternatively, it is the proportion of agreement after the chance agreement is removed from consideration" (Cohen, 1960). The coefficient is defined as

$$\kappa = \frac{P_o - P_c}{1 - P_c},$$

where P_o is the proportion of units for which the coders agreed (relative observed agreement among the raters) and P_c is the proportion of units for which agreement is expected by chance (chance-expected agreement).

Table 3
Contingency matrix.

		J_1			
		Category 1	Category 2	...	Category k
J_2	Category 1	$c_{1,1}$	$c_{2,1}$	...	$c_{k,1}$
	Category 2	$c_{1,2}$	$c_{2,2}$		...
	...				
	Category k	$c_{1,k}$	...		$c_{k,k}$

Table 4
Landis & Koch: interpretation of the value of κ.

Cohen's κ	Strength of Agreement
<0.0	Poor
0.00–0.20	Slight
0.21–0.40	Fair
0.41–0.60	Moderate
0.61–0.80	Substantial
0.81–1.00	Almost perfect

To calculate these proportions, we will use the so-called contingency matrix, as shown in Table 3. This is a square matrix of order the number of categories k. The (i, j)-entry, denoted $c_{i,j}$, is the number of times an item was assigned to the ith category by coder J_1 and to the jth category by coder J_2. In this way, the elements of the form $c_{i,i}$ are precisely the agreements in the evaluations.

From this contingency matrix, the observed agreement, P_o, and agreement by chance, P_c, are defined as is defined as

$$P_o = \frac{1}{m}\sum_{i=1}^{k} c_{i,i}, \quad P_c = \sum_{i=1}^{k} p_i,$$

where the probability of the ith category, p_i, is given by

$$p_i = \left(\frac{1}{m}\sum_{j=1}^{k} c_{i,j}\right)\left(\frac{1}{m}\sum_{j=1}^{k} c_{j,i}\right) = \frac{1}{m^2}\left(\sum_{j=1}^{k} c_{i,j}\right)\left(\sum_{j=1}^{k} c_{j,i}\right).$$

The coefficient is $\kappa = 0$ when the observed agreement is entirely due to a chance agreement. Greater-than-chance agreement corresponds to a positive value of κ and less-than-chance agreement corresponds to a negative value of κ. The maximum value of κ is 1, which occurs when (and only when) there is perfect agreement between the coders (Cohen, 1960). Landis and Koch, in Landis and Koch (1977), proposed the following table to evaluate intermediate values (Table 4).

As an example of application of this coefficient, let us come back to our example of Table 1. From the data, we compute the contingency matrix as shown in Table 5.

In this way, P_o is given by

$$P_o = \frac{1}{m}\left(c_{1,1} + c_{2,2}\right) = \frac{4+6}{15} = 0.667.$$

On the other hand, for P_c we have

$$p_1 = \frac{1}{15^2}(4+0)(4+5) = \frac{36}{225}, \quad p_2 = \frac{1}{15^2}(5+6)(0+6) = \frac{66}{225}.$$

Table 5
Contingency matrix for the data of Table 1.

		J_1		Total
		Y	N	
J_2	Y	4	5	9
	N	0	6	6
	Total	4	11	15

Hence $P_c = 0.453$. Therefore, we have

$$\kappa = \frac{P_o - P_c}{1 - P_c} = 0.391.$$

It is worth mentioning that, despite its simplicity, this coefficient has some intrinsic problems. On the one hand, it is limited to nominal data and two coders. On the other hand, it is difficult to interpret the result. Under various conditions, the κ statistic is affected by two paradoxes that return biased estimates of the statistic itself: (1) high levels of observer agreement with low κ values; (2) lack of predictability of changes in κ with changing marginals (Lantz and Nebenzahl, 1996). Some proposals for overcoming these paradoxes are described in Feinstein and Cicchetti (1990) and in Cicchetti and Feinstein (1990). According to Hayes and Krippendorff (2007): "κ is simply incommensurate with situations in which the reliability of data is the issue".

2.4. Fleiss' κ

Fleiss' κ is a generalization of Scott's π statistic to an arbitrary number, say $n \geq 2$, of raters $J_1, \ldots, J_n$. As above, we set m as the number of items to be coded and k as the number of categories (possible categorical ratings) under consideration. It is important to note that, while Cohen's κ assumes that the same two raters have rated a set of items, Fleiss' κ specifically allows that, although there are a fixed number of raters, different items may be rated by different individuals (Fleiss, 1971). Similarly to Scott's π, the coefficient is calculated using the formula

$$\kappa = \frac{P_o - P_e}{1 - P_e}.$$

For this coefficient, we will no longer focus on the contingency matrix but on the number of ratings. Therefore, given a category $1 \leq i \leq k$ and an item $1 \leq \beta \leq m$, we will denote by $n_{i,\beta}$ the number of raters who assigned the ith category to the β-th item.

In this case, for each item β and for each category i, we can compute the corresponding proportion of observations as

$$p_i = \frac{1}{nm} \sum_{\beta=1}^{m} n_{i,\beta},$$

$$\hat{p}_\beta = \frac{1}{n(n-1)} \sum_{i=1}^{k} n_{i,\beta}(n_{i,\beta} - 1) = \frac{1}{n(n-1)} \sum_{i=1}^{k} n_{i,\beta}^2 - n.$$

Recall that p_i is very similar to the one considered in Cohen's κ but, now, $\hat{p}_\beta$ counts the rate of pairs coder–coder that are in agreement (relative to the number of all possible coder–coder pairs).

In this way, the observed agreement, P_o, and the expected agreement, P_e, are the average of these quantities relative to the total number of possibilities, that is

$$P_o = \frac{1}{m} \sum_{\beta=1}^{m} \hat{p}_\beta, \quad P_e = \sum_{i=1}^{k} p_i^2.$$

As an example of application, consider again Table 1. Recall that, in our notation, the parameters of this example are $m = 15$ is the number of items (primary studies in our case), $n = 2$ is the number of coders and $k = 2$ is the number of nominal categories (Y and N in our example, which are categories 1 and 2, respectively).

Table 6
Count of the proportion of observations for Fleiss' κ for the example of Table 1.

$n_{i,\beta}$	#01	#02	#03	#04	#05	#06	#07	#08	#09
Y	1	1	0	0	0	1	2	1	0
N	1	1	2	2	2	1	0	1	2
$\hat{p}_\beta$	0	0	1	1	1	0	1	0	1

$n_{i,\beta}$	#10	#11	#12	#13	#14	#15	Total	p_i
Y	2	2	0	1	0	2	13	13/30 = 0.433
N	0	0	2	1	2	0	17	17/30 = 0.567
$\hat{p}_\beta$	1	1	1	0	1	1	10	

From these data, we form Table 6 with the calculation of the counting values $n_{i,\beta}$. In the second and third rows of this table, we indicate, for each item, the number of ratings it received for each of the possible categories (Y and N). For example, for item #01, J_1 voted it for the category N, while J_2 assigned it to the category Y and, thus, we have $n_{1,1} = n_{2,1} = 1$. On the other hand, for item #03 both coders assigned it the category N, so we have $n_{2,3} = 2$ and $n_{1,3} = 0$. In the last column of the table, we compute the observed percentages of each category, which give the results

$$p_1 = \frac{1}{2 \cdot 15}(1+1+0+0+0+1+2+1+0 + 2+2+0+1+0+2) = \frac{13}{30} = 0.433,$$

$$p_2 = \frac{1}{2 \cdot 15}(1+1+2+2+2+1+0+1+2 + 0+0+2+1+2+0) = \frac{17}{30} = 0.567.$$

Observe that, as expected, $p_1 + p_2 = 1$. Therefore, $P_e = 0.433^2 + 0.567^2 = 0.508$.

On the other hand, for the observed percentages per item, we have two types of results. First, if for the β-th item the two coders disagreed in their ratings, we have $\hat{p}_\beta = \frac{1}{2}(1 \cdot 0 + 1 \cdot 0) = 0$. However, if both coders agreed on their ratings (regardless of whether it was Y or N), we have $\hat{p}_\beta = \frac{1}{2}(2 \cdot 1 + 0 \cdot (-1)) = 1$. Their average is the observed agreement $P_o = \frac{1}{15}(5 \cdot 0 + 10 \cdot 1) = 0.667$. Therefore, the value of the Fleiss' κ coefficient is

$$\kappa = \frac{0.667 - 0.508}{1 - 0.508} = 0.322.$$

2.5. Krippendorff's α

The last coefficient that we will consider for measuring ICA is Krippendorff's α coefficient. Sections 3 and 4 are entirely devoted to the mathematical formulation of Krippendorff's α and its variants for content analysis. However, we believe that, for the convenience of the reader, it is worthy to introduce this coefficient here through a working example. The version we will discuss here corresponds to the universal α coefficient introduced in Krippendorff (1970) (see also Section 3), which only deals with simple codings as in the examples above. This is sometimes called binary α in the literature, but we reserve this name for a more involved version (see Section 4.2).

Again, we consider the data of Table 1, which corresponds to the simplest reliability data generated by two observers who assign one of the two available values to each of a common set of units of analysis (two observers, binary data). In this context,

Table 7
Observed coincidences matrix for the data of Table 1.

	Y	N	
Y	$o_{1,1} = 8$	$o_{1,2} = 5$	$t_1 = 13$
N	$o_{1,2} = 5$	$o_{2,2} = 12$	$t_2 = 17$
	$t_1 = 13$	$t_2 = 17$	$t = 30$

Table 8
Expected coincidences matrix for the data of Table 1.

	Y	N	
Y	$e_{1,1} = 5.38$	$e_{1,2} = 7.62$	$t_1 = 13$
N	$e_{1,2} = 7.62$	$e_{2,2} = 9.38$	$t_2 = 17$
	$t_1 = 13$	$t_2 = 17$	$t = 30$

this table is called the reliability data matrix. From this table, we construct the so-called matrix of observed coincidences, as shown in Table 7. This is a square matrix of order the number of possible categories (hence, a 2×2 matrix in our case since we only deal with the categories Y and N).

The way in which this table is built is the following. First, in Table 1 you need to count in Table 1 the number of pairs (Y, Y). In this case, 4 items received two Y from the coders (#07, #10, #11 and #15). However, the observed coincidences matrix counts *ordered pairs* of judgments, and in the previous count J_1 always shows up first and J_2 appears second. Therefore, we need to multiply this result by 2, obtaining a total count of 8 that is written down in the (Y, Y) entry of the observed coincidences matrix, denoted $o_{1,1}$. In the same spirit, the entry $o_{2,2}$ of the matrix corresponds to the $2 \cdot 6 = 12$ ordered pairs of responses (N, N). In addition, the anti-diagonal entries of the matrix, $o_{1,2}$ and $o_{2,1}$, correspond to responses of the form (Y, N) and (N, Y). There are 5 items in which we got a disagreement (#01, #02, #06, #08 and #14), so, as ordered pairs of responses, there are 5 pairs of responses (Y, N) and 5 pairs of responses (N, Y), which are written down in the observed coincidences matrix. Finally, the marginal data, t_1 and t_2, are the sums of the values of the rows and columns, and $t = t_1 + t_2$ is twice the number of items. Observe that, by construction, the observed coincidences matrix is symmetric.

In this way, the observed agreement is given by

$$P_o = \frac{o_{1,1}}{t} + \frac{o_{2,2}}{t} = \frac{8}{30} + \frac{12}{30} = 0.67.$$

On the other hand, as in the previous methods, we need to compare these observed coincidences with the expected coincidences by chance. This information is collected in the so-called expected coincidences matrix, as shown in Table 8. The entries in this matrix, $e_{i,j}$, measure the probability of getting an ordered response (i, j) entirely by chance. In our case, the expected coincidences are given by

$$e_{1,1} = \frac{t_1(t_1 - 1)}{t - 1} = \frac{13 \cdot (13 - 1)}{30 - 1} = 5.38,$$

$$e_{2,2} = \frac{t_2(t_2 - 1)}{t - 1} = \frac{17 \cdot (17 - 1)}{30 - 1} = 9.38,$$

$$e_{1,2} = e_{2,1} = \frac{t_1 t_2}{t - 1} = \frac{13 \cdot 17}{30 - 1} = 7.62.$$

Therefore, the expected agreement is

$$P_e = \frac{e_{1,1}}{t} + \frac{e_{2,2}}{t} = \frac{5.35}{30} + \frac{9.38}{30} = 0.49.$$

Thus, using the same formula for the ICA coefficient as in Section 2.2, we find that Krippendorff's α is given by

$$\alpha = \frac{P_o - P_e}{1 - P_e} = \frac{0.67 - 0.49}{1 - 0.49} = 0.343.$$

As a final remark, in the context of Krippendorff's α, it is customary to use the equivalent formulation

$$\alpha = 1 - \frac{D_o}{D_e},$$

where $D_o = o_{1,2}$ is the observed disagreement and $D_e = e_{1,2}$ is the expected disagreement.

3. The universal Krippendorff's α coefficient

Krippendorff's α coefficient is one of the most widely used coefficients for measuring Inter-Coder Agreement in content analysis. As we mentioned in Section 2.5, this coefficient addresses many of the flaws that Cohen's κ and Fleiss' κ suffer. For a more detailed exposition comparing these coefficients see Hayes and Krippendorff (2007) and for a historical description of this coefficient, please check Krippendorff (2018).

In this section, we present a new probabilistic framework that underpins Krippendorff's α coefficient. For this purpose, we introduce a novel interpretation that unifies the different variants of the α coefficient presented in the literature (see, for instance, Krippendorff, 1970, 1995; Hayes and Krippendorff, 2007; Yang and Green, 2011; Gwet, 2011; De Swert, 2012; Krippendorff et al., 2016; Krippendorff, 2018). These variants are usually presented as unrelated and through an ad-hoc formulation for each problem. This turns the use of Krippendorff's α for the unfamiliar researcher confusing and unmotivated. For this reason, it is necessary to provide a common framework in which precise interpretations and comparisons of this metric can be made. Subsequently, in Section 4, we will provide descriptions of these variants in terms of this universal α coefficient. The present formulation is an extension of our previous work (Díaz et al., 2021) towards a uniform formulation.

Suppose that we are dealing with $n \geq 2$ different judges, also referred to as coders, denoted by $J_1, \ldots, J_n$; as well as with a collection of $m \geq 1$ items to be judged, also referred to as quotations, denoted $I_1, \ldots, I_m$. We fix a set of $k \geq 1$ admissible 'meta-codes', called labels, say $\Lambda = \{l_1, \ldots, l_k\}$.

The task of each of the coders J_α is to assign, to each item I_β, a collection (maybe empty) of labels from Λ. Therefore, as a by-product of the evaluation process, we get a set $\Omega = \{\omega_{\alpha,\beta}\}$, for $1 \leq \alpha \leq n$ and $1 \leq \beta \leq m$, where $\omega_{\alpha,\beta} \subseteq \Lambda$ is the set of labels that the coder J_α assigned to the item I_β. Recall that $\omega_{\alpha,\beta}$ is not a multiset, so every label appears in $\omega_{\alpha,\beta}$ at most once. Moreover, notice that multi-evaluations are now allowed, that is, a coder may associate more than a label to an item. This translates to the fact that $\omega_{\alpha,\beta}$ may be empty (meaning that J_α did not assign any label to I_β), it may have a single element (meaning that J_α assigned only one label), or it may have more than an element (meaning that J_α chose several labels for I_β).

From the collection of responses Ω, we can count the number of observed pairs of responses. For that, fix $1 \leq i, j \leq k$ and set

$$o_{i,j} = \left|\left\{(\omega_{\alpha,\beta}, \omega_{\alpha',\beta}) \in \Omega \times \Omega \;\middle|\; \alpha' \neq \alpha, l_i \in \omega_{\alpha,\beta} \text{ and } l_j \in \omega_{\alpha',\beta}\right\}\right|.$$

In other words, $o_{i,j}$ counts the number of (ordered) pairs of responses of the form $(\omega_{\alpha,\beta}, \omega_{\alpha',\beta}) \in \Omega \times \Omega$ that two different coders J_α and $J_{\alpha'}$ gave to the same item I_β and such that J_α included l_i in his response and $J_{\alpha'}$ included l_j in his response. In the notation of Section 2.3, in the case that $n = 2$ (two coders), we have $o_{i,j} = c_{i,j} + c_{j,i}$.

Remark 1. Suppose that there exists an item I_β that was judged by a single coder, say J_α. The other coders, $J_{\alpha'}$ for $\alpha' \neq \alpha$, did not vote for it, so $\omega_{\alpha',\beta} = \emptyset$. Then, this item I_β does not contribute to the calculation of $o_{i,j}$ as there is no other judgement to which $\omega_{\alpha,\beta}$ can be paired. Hence, from the point of view of Krippendorff's α, I_β is not taken into account. This causes some strange behaviors in the coefficients of Section 4 that may seem counterintuitive.

From these counts, we construct the matrix of observed coincidences as $M_o = \left(o_{i,j}\right)_{i,j=1}^{k}$. By its very construction, M_o is a symmetric matrix. From this matrix we set $t_k = \sum_{j=1}^{k} o_{k,j}$, which is (twice) the total number of times to which the label $l_k \in \Lambda$ was assigned by any coder. Observe that $t = \sum_{k=1}^{k} t_k$ is the total number of judgments. In the case that each coder evaluates each item with a single non-empty label, we have $t = nm$.

On the other hand, we can construct the matrix of expected coincidences, $M_e = \left(e_{i,j}\right)_{i,j=1}^{k}$, where

$$e_{i,j} = \begin{cases} \frac{t_i}{t}\frac{t_j}{t-1}t = \frac{t_i t_j}{t-1} & \text{if } i \neq j, \\ \frac{t_i}{t}\frac{t_i-1}{t-1}t = \frac{t_i(t_i-1)}{t-1} & \text{if } i = j. \end{cases}$$

The value of $e_{i,j}$ might be thought of as the average number of times that we expect to find a pair (l_i, l_j), when the frequency of the label l_i is estimated from the sample as t_i/t. It is analogous to the value of the proportion $\hat{p}_\beta$ in Section 2.4. Again, M_e is a symmetric matrix.

Finally, let us fix a pseudo-metric $\delta : \Lambda \times \Lambda \to [0, \infty) \subseteq \mathbf{R}$, i.e. a symmetric function satisfying the triangle inequality and with $\delta(l_i, l_i) = 0$ for any $l_i \in \Lambda$ (recall that this is only a pseudo-metric since different labels at distance zero are allowed). This metric is given by the semantics of the analyzed problem, and thus, it is part of the data used for quantifying the agreement. The value $\delta(l_i, l_j)$ should be seen as a measure of how similar the labels l_i and l_j are. Some common choices of this metric are the following.

- The discrete metric. It is given by $\delta(l_i, l_j) = 0$ if $i = j$ and $\delta(l_i, l_j) = 1$ otherwise. The discrete metric means that all labels are equally separated and is the one that will be used throughout this paper.
- The Euclidean distance. If the labels l_i are vectors (for example, if l_i is a collection of real values measuring some features), we can take $\delta(l_i, l_j) = \|l_i - l_j\|$, the norm of the difference vector $l_i - l_j$. In the particular case where l_i are real values, this metric reduces to the absolute distance.
- The angular metric. If the labels l_i take values in an angular measure (for example, the deviation in some experiment), we can take $\delta(l_i, l_j) = \sin^2(l_i - l_j)$. Observe that $\delta(l_i, l_j) = 0$ if l_i and l_j are opposed, showing that δ is only a pseudo-metric.

For subtler metrics that may be used for extracting more semantic information from the data, see Krippendorff et al. (2016).

From these computations, we define the observed disagreement D_o and the expected disagreement D_e, as

$$D_o = \sum_{i=1}^{k}\sum_{j=1}^{k} o_{i,j}\delta(l_i, l_j), \qquad D_e = \sum_{i=1}^{k}\sum_{j=1}^{k} e_{i,j}\delta(l_i, l_j). \tag{1}$$

These quantities measure the degree of disagreement that is observed from Ω and the degree of disagreement that might be expected by judging randomly (i.e. by chance), respectively.

Remark 2. If we take δ as the discrete metric, we have another interpretation of the disagreement. Observe that, in this case, since $\delta(l_i, l_j) = 0$ for $i \neq j$ and $\delta(l_i, l_i) = 1$, we can write the disagreements as

$$D_o = \sum_{i\neq j} o_{i,j} = t - \sum_{i=1}^{k} o_{i,i}, \qquad D_e = \sum_{i\neq j} e_{i,j} = t - \sum_{i=1}^{k} e_{i,i}.$$

The quantity $P_o = \sum_{i=1}^{k} o_{i,i}$ (resp. $P_e = \sum_{i=1}^{k} e_{i,i}$) can be understood as the observed (resp. expected) agreement between the coders. In the same vein, $t = \sum_{i,j=1}^{k} o_{i,j} = \sum_{i,j=1}^{k} e_{i,j}$ may be seen as the maximum achievable agreement. Therefore, in this context, the disagreement D_o (resp. D_e) is actually the difference between the maximum possible agreement and the observed (resp. expected) agreement.

From these data, Krippendorff's α coefficient is defined as

$$\alpha = \alpha(\Omega) = 1 - \frac{D_o}{D_e}.$$

From this formula, observe the following limiting values:

- $\alpha = 1$ is equivalent to $D_o = 0$ or, in other words, it means that there exists perfect agreement in the judgments among the coders.
- $\alpha = 0$ is equivalent to $D_o = D_e$, which means that the agreement observed between the judgments is entirely due to chance.

In this way, Krippendorff's α can be interpreted as a measure of the degree of agreement that is achieved out of chance. The bigger α is, the better agreement is observed. A common rule-of-thumb in the literature (Krippendorff, 2018) is that $\alpha \geq 0.667$ is the minimal threshold required to draw conclusions from the data. For $\alpha \geq 0.80$, we can consider that there exists statistical evidence of reliability in the evaluations. Apart from these considerations, there are doubts in the community that more partitioned interpretations, such as the one of Landis & Koch of Table 4, are valid in this context (see Krippendorff, 2018).

Remark 3. Observe that $\alpha < 0$ can only be achieved if $D_o > D_e$, which means that there is even more disagreement than the one that could be expected by chance. This implies that the coders are, consistently, issuing different judgments for the same items. Thus, it evidences that there exists an agreement between the coders to not agree, that is, to fake the evaluations. On the other hand, as long as the metric δ is non-negative, $D_o \geq 0$ and thus $\alpha \leq 1$.

Remark 4. The observed and expected coincidences, $o_{i,j}$ and $e_{i,j}$, do not depend on the particular coders under consideration. All the coders are mathematically treated on an equal footing. In particular, this allows us to split the matter under analysis into several batches that will be coded by different teams of coders. Even more, each of these teams may have a different number of coders. In this scenario, the previous formulation of the α coefficient still works, and the resulting coefficient measures a kind of weighted average of the agreement of each batch.

This observation may seem particularly attractive for researchers because it allows them to parallelize the coding process with several teams of coders. However, in practice, this parallelization is discouraged. As we will see in Section 5, after each round of coding, the codebook designer provides feedback to the coders on precise definitions and better bound the limits of application of the codes. This typically leads to a remarkable improvement in the observed agreement. Nevertheless, if we parallelize the coding process, the interpretation of the codebook of the different coding teams may be different and the solutions implemented to address the interpretation issues might be opposed. In this way, the improvement of the ICA stagnates and subsequent refinements of the codebook do not lead to better results.

4. Theoretical framework: Semantic domains and variants of the α coefficient

The setting described in Sections 2 and 3 only addresses the problem of evaluating the agreement when the judges use la-

bels to code data. This might be too restrictive for the purposes of qualitative data analysis, where the codes typically form a two-layer structure: semantic domains that encompass broad interrelated concepts and codes within the domains that capture subtler details. This is, for instance, the setting considered by qualitative analysis tools like the Atlas.ti software (ATLAS.ti Scientific Software Development GmbH, 2020).

In this section, we describe a more general framework that enables evaluating reliability in this two-layer structure. However, this more involved setting also leads to more aspects of reliability that must be measured. It is not the same to evaluate reliability in the choice of the semantic domain to be applied as in the chosen codes within a particular domain or the agreement achieved when distinguishing between relevant and irrelevant matter. To assess the reliability of these aspects, several variants of Krippendorff's α have been proposed in the literature (up to 10 are mentioned in Krippendorff, 2018, Section 12.2.3). In this vein, we explain some of these variants and how they can be reduced to the universal α coefficient of Section 3 after an algorithmic translation by relabeling codes. As Section 3, this framework is an extension of the paper (Díaz et al., 2021).

To be precise, in coding we usually need to consider a two-layers setting as follows. First, we have a collection of $s > 1$ semantic domains, $S_1, \ldots, S_s$. A semantic domain defines a space of distinct concepts that share a common meaning (say, S_i might be colors, brands, feelings, etc.). Subsequently, each semantic domain embraces mutually exclusive concepts indicated by a code. Hence, for $1 \leq i \leq s$, the domain S_i decomposes into $r_i \geq 1$ codes, which we denote by $C^i_1, \ldots, C^i_{r_i}$. For design consistency, these semantic domains must be logically and conceptually independent. This principle translates into the fact that there exist no shared codes between different semantic domains and that two codes within the same semantic domain cannot be applied at the same time by a coder to the same quotation.

Now, the data under analysis (e.g. scientific literature, newspapers, videos, interviews) are chopped into items, which in this context are known as *quotations*, which represent meaningful parts of the data on their own. The decomposition may be decided by each of the coders (so different coders may have different quotations) or it may be pre-established (for instance, by the codebook creator or the designer of the ICA study). In the latter case, all the coders share the same quotations, so they cannot modify their limits and they should evaluate each quotation as a block. In order to enlighten the notation, we shall suppose that we are dealing with this case of pre-established quotations. Indeed, from a mathematical point of view, the former case can be reduced to this version by refining the data subdivision of each coder to get a common decomposition into the same pieces.

Therefore, we will suppose that the data is previously decomposed into $m \geq 1$ items or quotations, $I_1, \ldots, I_m$. Observe that the union of all the quotations must be the whole matter so, in particular, irrelevant matter is also included as quotations. Now, each of the coders J_α, $1 \leq \alpha \leq n$, evaluates the quotations I_β, $1 \leq i \leq m$, assigning to I_β any number of semantic domains, and, for each chosen semantic domain, one and only one code. No semantic domain may be assigned in the case where the coder considers that I_β is irrelevant matter, and several domains can be applied to I_β by the same coder.

Hence, as byproduct of the evaluation process, we obtain a collection of sets $\Sigma = \{\sigma_{\alpha,\beta}\}$, for $1 \leq \alpha \leq n$ and $1 \leq \beta \leq m$. Here, $\sigma_{\alpha,\beta} = \left\{C^{i_1}_{j_1}, \ldots, C^{i_p}_{j_p}\right\}$ is the collection of codes that the coder J_α assigned to the quotation I_β. The exclusion principle of codes within the semantic domain means that the collection of chosen semantic domains $i_1, \ldots, i_p$ contains no repetitions.

Remark 5. To be precise, as proposed in Krippendorff (1995), when dealing with a continuum of matter, each of the quotations must be weighted by its length in the observed and expected coincidences matrices. This length is defined as the amount of atomic units the quotation has (say characters in a text or seconds in a video). In this way, (dis)agreements in long quotations are more significant than (dis)agreements in short quotations. This can be easily incorporated into our setting just by refining the data decomposition to the level of units. In this way, we create new quotations that are the length of an atomic unit. Each new atomic quotation is judged with the same evaluations as the old larger quotation. In the coefficients introduced below, this idea has the mathematical effect that, in the sums of Eq. (1), each old quotation appears as many times as the atomic units it contains, which is the length of such quotation. Therefore, in this manner, the version explained here computes the same coefficient as in Krippendorff (1995).

To quantify the degree of agreement achieved by the coders in the evaluations Σ, several variants of Krippendorff's α are proposed in the literature (Krippendorff et al., 2016; Krippendorff, 2018). Some of the most useful for case studies are the following variants.

- The coefficient α^{gl}_{binary}: This is a global measure. It quantifies the agreement of the coders when identifying relevant matter (quotations that deserve to be coded) and irrelevant matter (part of the corpus that is not coded).
- The coefficient α_{binary}: This coefficient is computed on a specific semantic domain S_i. It is a measure of the degree of agreement that coders achieve when choosing to apply a semantic domain S_i or not.
- The coefficient *cu-α*: This coefficient is computed on a semantic domain S_i. It indicates the degree of agreement to which coders identify codes within S_i.
- The coefficient *Cu-α*: This is a global measure of the goodness of the partition into semantic domains. *Cu-α* measures the degree of reliability in the decision of applying the different semantic domains, independently of the chosen code.

Before diving into the detailed formulation, let us work out an illustrative example. Fig. 1 shows an example of the use of these coefficients. Let us consider three semantic domains, whose their respective codes being as follows

$$S_1 = \{C_{11}, C_{12}\}, \quad S_2 = \{C_{21}, C_{22}\}, \quad S_3 = \{C_{31}, C_{32}\}.$$

The two coders, J_1 and J_2, assign codes to four quotations as shown in Fig. 1(a). We created a graphical metaphor so that each coder, each semantic domain, and each code are represented as shown in Fig. 1(b). Each coder is represented by a shape, so that J_1 is represented by triangles and J_2 by circles. Each domain is represented by a color, so that S_1 is red, S_2 is blue, and S_3 is green. Each code within the same semantic domain is represented by a filling, so that C_{i1} codes are represented by a solid filling and C_{i2} codes are represented by a dashed filling.

The coefficient α_{binary} is calculated per domain (i.e. S_1 red, S_2 blue, S_3 green) and analyzes whether the coders assigned or not a domain—independently of the code—to the quotations (see Fig. 1(c)). Notice that we only focus on the presence or absence of a semantic domain by quotation, so Fig. 1(c) only takes into account the color. Now, the α_{binary} coefficient measures the agreement that the coders achieved in assigning the same color to the same quotation. The bigger the coefficient, the better the agreement. In this way, we get total agreement ($\alpha_{binary} = 1$) for S_2 as both coders assigned this domain (blue) to the second quotation and the absence of this domain in the rest of quotations.

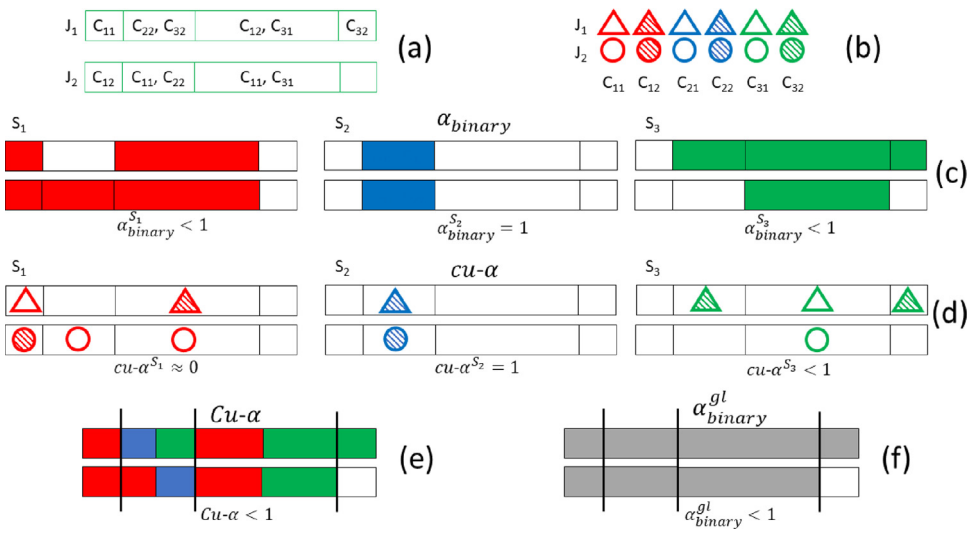

Fig. 1. Illustrative example of coefficients.

Table 9
Equivalence of notations between the variants of the α coefficient.
Source: Notation for the Atlas software has been extracted from ATLAS.ti Scientific Software Development GmbH (2020) and Friese (2019).

Name	Krippendorff (2018)	Atlas	This paper
Global binary α	${}_{u}\alpha$	alpha-binary (global)	α^{gl}_{binary}
Binary α per domain	${}_{\|cu}\alpha$, ${}_{\|u}\alpha$	alpha-binary (domain)	α_{binary}
cu-α	${}_{(s)u}\alpha$, ${}_{(k)u}\alpha$	cu-alpha	*cu-α*
Cu-α	${}_{Su}\alpha$	Cu-alpha	*Cu-α*

On the other hand, $\alpha_{binary} < 1$ for S_1 as J_1 assigned this domain (red) to quotations 1 and 3 while J_2 assigned it to quotations 1, 2 and 3, leading to a disagreement in quotation 2.

The coefficient *cu-α* is also calculated per domain (i.e. S_1 red, S_2 blue, S_3 green), but it measures the agreement attained when applying the codes of that domain. In other words, given a domain S_i, this coefficient analyzes whether the coders assigned the same codes of S_i (i.e. the same fills) to the quotations or not. In this way, as shown in Fig. 1(d), it only focuses on the applied fills to each quotation. In particular, observe that *cu-α* $= 1$ for S_2 since both coders assigned the same code to the second quotation and no code from this domain to the rest of the quotations, i.e. total agreement. Also, notice that *cu-α* < 1 for S_3 as the coders assigned the same code of S_3 to the third quotation but they did not assign the same codes of S_3 to the rest of the quotations. Finally, observe that *cu-α* for S_1 is very small (near to zero) since the coders achieve no agreement on the chosen codes.

Regarding the global coefficients, the coefficient *Cu-α* analyzes all the domains as a whole, but it does not take into account the codes within each domain. In this way, in Fig. 1(e), we color each segment with the colors corresponding to the applied semantic domain (regardless of the particular code used). From this chromatic representation, *Cu-α* measures the agreement in applying these colors globally between the coders. In particular, notice that *Cu-α* < 1 as both coders assigned the same domain S_1 to the first quotations and the domains S_1 and S_3 to the third quotation, but they did not assign the same domains in the second and fourth quotations.

Finally, the α^{gl}_{binary} coefficient measures the agreement in the selection of relevant matter, as shown in Fig. 1(f). In this case, both coders recognized the first three segments as relevant (they were coded), as highlighted in gray in the figure. However, J_2 considered that the fourth quotation was irrelevant (it was not coded), as marked in white, and J_1 marked it as relevant, in gray. In this way, we have that $\alpha^{gl}_{binary} < 1$.

Remark 6. There is no consensus in the literature on the notation used for the different variants of the α coefficients. This leads to inter-operation malfunction when we try to change from one scientific source to another or to use a different software to instrumentalize the computation of the ICA coefficients. In particular, Atlas.ti, a leading software in qualitative research, uses a different nomenclature for the variants than the original work by Krippendorff and a different notation than ours.

For the convenience of the reader, in Table 9 we include a comparison between the original Krippendorff's notation (Krippendorff, 2018), the Atlas.ti notation for the α coefficient and the notation used in this paper. Note that one of the goals of the present work is precisely to introduce a unifying framework in which all these coefficients emerge as mere variants of a single statistical measure.

4.1. The coefficient α^{gl}_{binary}

The first variation of Krippendorff's α coefficient that we consider is the α^{gl}_{binary} coefficient. It is a global measure that summarizes the agreement of the coders for recognizing relevant parts of the matter. For computing it, we consider a set of labels having only two labels, that semantically represent 'recognized as relevant' (1) and 'not recognized as relevant' (0). Hence, we take

$$\Lambda = \{1, 0\}.$$

Now, using the entire set of evaluations Σ, we create a new labeling $\Omega^{gl}_{bin} = \{\omega_{\alpha,\beta}\}$ as follows. Let $1 \leq \alpha \leq n$ and $1 \leq \beta \leq m$. We set $\omega_{\alpha,\beta} = \{1\}$ if the coder J_α assigned some code to the quotation I_β (i.e. if $\sigma_{\alpha,\beta} \neq \emptyset$) and $\omega_{\alpha,\beta} = \{0\}$ otherwise (i.e. if J_α did not code I_β, that is $\sigma_{\alpha,\beta} = \emptyset$). From this set of evaluations, $\Omega^{gl}_{bin} = \{\omega_{\alpha,\beta}\}$, α^{gl}_{binary} is given as

$$\alpha^{gl}_{binary} = \alpha(\Omega^{gl}_{bin}).$$

Therefore, α^{gl}_{binary} measures the degree of agreement that the coders achieved when recognizing relevant parts, that is coded parts and irrelevant matter. A high value of α^{gl}_{binary} may be interpreted as that the matter is well structured and it is relatively easy to detect and isolate the relevant parts of information.

Remark 7. In many studies (for instance in case studies in software engineering), it is customary that a researcher pre-processes the raw data to be analyzed, say by transcribing it or by writing it down into an ICA software. In that case, usually this pre-processor selects the parts that must be analyzed and chops the matter into quotations before the starting of the judgement process. In this way, the coders are required to code these pre-selected parts, so that they no longer chop the matter by themselves and they code all the quotations. Hence, we always get that $\alpha^{gl}_{binary} = 1$, since the evaluation protocol forces the coders to consider as relevant matter the selected parts by the pre-processor. Therefore, in these scenarios, the α^{gl}_{binary} coefficient is not useful for providing reliability on the evaluations and other coefficients of the α family are required.

4.2. The coefficient α_{binary}

The second variation of the Krippendorff's α coefficient is the so-called α_{binary} coefficient. This is a coefficient that must be computed on a specific semantic domain. Hence, let us fix a semantic domain S_i for some fixed i with $1 \leq i \leq s$. As above, the set of labels will have only two labels that semantically represent 'voted S_i' (label 1) and 'did not vote S_i' (label 0). Hence, we take

$$\Lambda = \{1, 0\}.$$

For the assignment of labels to items, the rule is as follows. For $1 \leq \alpha \leq n$ and $1 \leq \beta \leq m$, we set $\omega_{\alpha,\beta} = \{1\}$ if the coder J_α assigned some code of S_i to the quotation I_β (i.e. if $C^i_j \in \sigma_{\alpha,\beta}$ for some $1 \leq j \leq r_i$) and $\omega_{\alpha,\beta} = \{0\}$ otherwise. Observe that, in particular, $\omega_{\alpha,\beta} = \{0\}$ if J_α considered that I_β was irrelevant matter. From this set of evaluations, $\Omega^{S_i}_{binary} = \left\{\omega_{\alpha,\beta}\right\}$, $\alpha^{S_i}_{binary}$ is given as

$$\alpha^{S_i}_{binary} = \alpha(\Omega^{S_i}_{binary}).$$

In this way, the coefficient $\alpha^{S_i}_{binary}$ can be seen as a measure of the degree of agreement the coders achieved when choosing to apply the semantic domain S_i or not. A high value of $\alpha^{S_i}_{binary}$ is interpreted as a piece of evidence that the domain S_i is clearly stated, its boundaries are well defined, and thus the decision of applying it or not is near to be deterministic. However, observe that it does not measure the degree of agreement in the application of the different codes within the domain S_i. Hence, it may occur that the boundaries of the domain S_i are clearly defined but the inner codes are not well chosen. This is not a task of the $\alpha^{S_i}_{binary}$ coefficient, but of the $cu\text{-}\alpha^{S_i}$ coefficient explained below.

Remark 8. By the definition of α_{binary}, in line with the implementation in ICA supporting software like Atlas.ti (ATLAS.ti Scientific Software Development GmbH, 2020), the irrelevant matter plays a role in the computation. As mentioned above, all the matter that was evaluated as irrelevant (i.e. was not coded) is labeled with {0}. In particular, a large corpus with only a few sparse short quotations may distort the value of α_{binary}.

4.3. The coefficient cu-α

Another variation of the Krippendorff's α coefficient is the so-called $cu\text{-}\alpha$ coefficient. As in the previous variation, this coefficient is calculated per semantic domain, say S_i for some $1 \leq i \leq s$. Suppose that this semantic domain contains codes $C^i_1, \ldots, C^i_r$. The collection of labels is now a set

$$\Lambda = \{\mathcal{C}_1, \ldots, \mathcal{C}_r\}.$$

Semantically, they are labels that represent the codes of the chosen domain S_i.

For the assignment of labels to items, the rule is as follows. For $1 \leq \alpha \leq n$ and $1 \leq \beta \leq m$, we set $\omega_{\alpha,\beta} = \mathcal{C}_k$ if the coder J_α assigned the code C^i_k of S_i to the item (quotation) I_β. Recall that, from the exclusion principle for codes within a semantic domain, the coder J_α applied at most one code from S_i to I_β. If the coder J_α did not apply any code of S_i to I_β, we set $\omega_{\alpha,\beta} = \emptyset$. From this set of judgments $\Omega^{S_i}_{cu} = \left\{\omega_{\alpha,\beta}\right\}$, $cu\text{-}\alpha^{S_i}$ is given as

$$cu\text{-}\alpha^{S_i} = \alpha(\Omega^{S_i}_{cu}).$$

Remark 9. As explained in Remark 1, for the computation of the observed and expected coincidence matrices, only items that received at least two evaluations with codes of S_i from two different coders count. In particular, if a quotation is not evaluated by any coder (irrelevant matter), received evaluations for other domains but not for S_i (matter that does not correspond to the chosen domain) or only one coder assigned to it a code from S_i (singled voted), the quotation does not play a role in $cu\text{-}\alpha$. This limitation might seem a bit cumbersome, but it could be explained by arguing that the presence/absence of S_i is measured by $\alpha^{S_i}_{binary}$ so it will be redundant to take it into account for $cu\text{-}\alpha^{S_i}$ too.

4.4. The coefficient Cu-α

The last variation of Krippendorff's α coefficient that we consider in this study is the so-called $Cu\text{-}\alpha$ coefficient. In contrast to the previous coefficients, this is a global measure of the goodness of partition into semantic domains. Suppose that our codebook determines semantic domains $S_1, \ldots, S_s$. In this case, the collection of labels is the set

$$\Lambda = \{\mathcal{S}_1, \ldots, \mathcal{S}_s\}.$$

Semantically, they are labels that represent the semantic domains of our codebook.

We assign labels to the items as follows. Let $1 \leq \alpha \leq n$ and $1 \leq \beta \leq m$. Then, if $\sigma_{\alpha,\beta} = \left\{C^{i_1}_{j_1}, \ldots, C^{i_p}_{j_p}\right\}$, we set $\omega_{\alpha,\beta} = \left\{\mathcal{S}_{i_1}, \ldots, \mathcal{S}_{i_p}\right\}$. In other words, we label I_β with the labels corresponding to the semantic domains chosen by the coder J_α for this item, independently of the particular code. Observe that this is the first case in which the final evaluation Ω might be multivalued. From this set of judgments, $\Omega_{Cu} = \left\{\omega_{\alpha,\beta}\right\}$, $Cu\text{-}\alpha$ is given as

$$Cu\text{-}\alpha = \alpha(\Omega_{Cu}).$$

In this way, $Cu\text{-}\alpha$ measures the degree of reliability in the decision to apply the different semantic domains, independently of the particular chosen code. Therefore, it is a global measure that quantifies the logical independence of the semantic domains and the ability of the coders to look at the big picture of the matter, only from the point of view of semantic domains.

5. Inter-Coder Agreement (ICA) in practice

In this section, we shall briefly describe how to apply the previously introduced concepts of ICA to assess the reliability in the findings of a qualitative research in software engineering.

As a guide, we shall use an excerpt from previous research by the authors in the domain of DevOps (Díaz et al., 2021). The considered example is an exploratory study to characterize the reasons why companies move to DevOps and what results they expect to obtain when adopting the DevOps culture (Leite et al., 2019). This exploratory case study is based on interviews with software practitioners from 30 multinational software-intensive

companies. The study has been conducted according to the guidelines for conducting qualitative research in software engineering proposed by Wohlin et al. (2012). Observe that the conclusions of this previous research do not represent the goal of the present work: The case study will only act as an example to illustrate the ICA techniques. A more thorough explanation of this case study can be found in Díaz et al. (2021). Moreover, a complete description of the process, including a detailed tutorial on how to instrument the computations of the α coefficients with the software Atlas.ti v9 (ATLAS.ti Scientific Software Development GmbH, 2020) can be found in https://blogs.upm.es/qualitativeresearch/atlas-ti-for-inter-coder-agreement-ica-a-tutorial/.

In Fig. 2 we show, through a UML activity diagram (Rumbaugh et al., 1999), the different stages that comprise the above-mentioned study. In the following exposition, each step of the analysis is accompanied by a brief explanation of the underlying qualitative research methodology that the authors carried out.

5.1. Set research objectives

The first step needed for conducting an exploratory study is to define the objective of the prospective work, the so-called research question (RQ). These objectives must be clearly stated, and the boundaries of each research question should be undoubtedly demarcated.

In the case study of the running example presented in this paper, we propose one research question related to the implications of instilling a DevOps culture in a company, which is the main concern of the analysis.

RQ: *What problems do companies try to solve by implementing DevOps?*

All subsequent data collection, analysis, and discussion must be oriented to address the posed research questions. In this setting, ICA techniques will allow us to straighten our conclusions by assessing their reliability.

5.2. Collect data

The next step in the research is to collect the empirical evidence needed for understanding the phenomenon under study. Data are the only window that researchers have to the object of research, so getting high-quality data typically leads to good research. As a rule-of-thumb, the better the data, the more precise the conclusions can be drawn.

There are two main methods for collecting information in qualitative analysis, and both are particularly useful in software engineering: questionnaires and interviews (Wohlin et al., 2012). Usually, questionnaires are easier to issue, since they can be handed by email or by web pages that can be accessed anytime and anywhere. On the other hand, interviews tend to gather a more complete picture of the phenomenon under study since there exists an active interaction between interviewer and interviewee, typically face-to-face. In this way, interviews are usually better suited for case studies since they allow the researcher to modify the questions to be asked on the fly to emphasize the key points under analysis. As a drawback, typically the number of answers that can be obtained through a questionnaire is much larger than the number of interviews that can be conducted, but the latter usually leads to higher quality data.

In the study considered in this paper, the data collection method was semi-structured interviews with software practitioners from 30 companies. The interviews were conducted face-to-face, using the Spanish language, and the audio was recorded with the permission of the participants, transcribed for the purpose of data analysis, and reviewed by respondents.

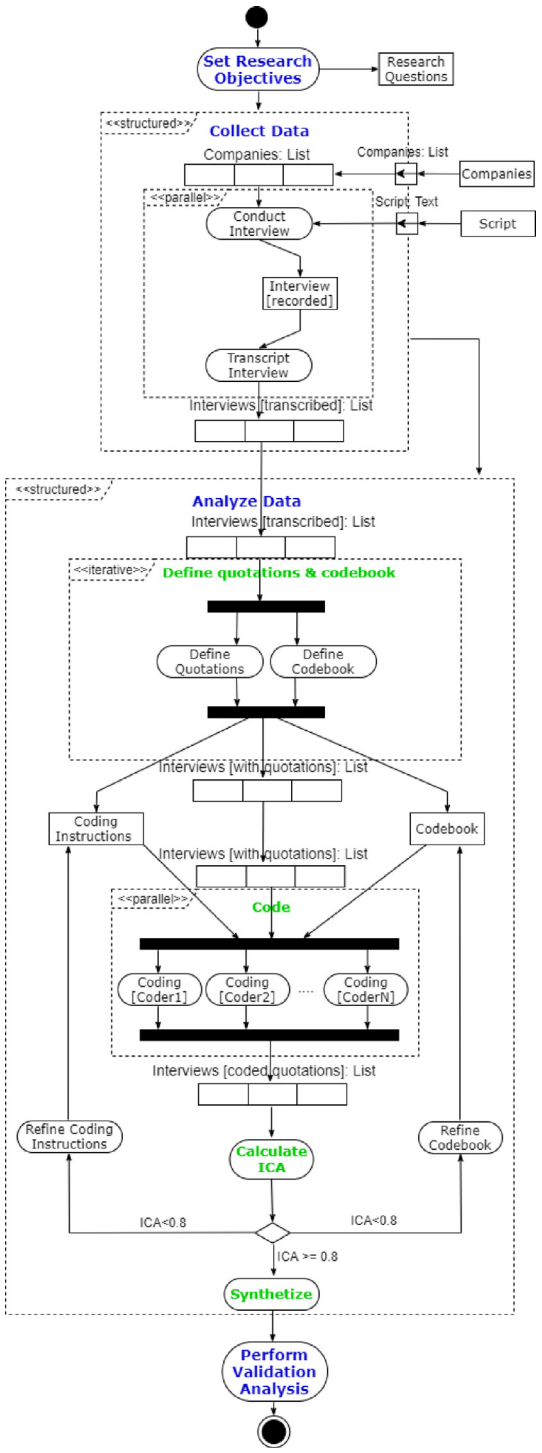

Fig. 2. Phases of case study research that involves qualitative data analysis in software engineering.

5.3. Analyze data

This is the most important phase of the study. In this step, researchers turn the raw data into structured and logically interconnected conclusions. However, because of its creative component, it is the least straightforward phase in the cycle.

To help researchers analyze the data and draw conclusions, there exist several methods for qualitative data analysis that can be followed. In the DevOps exploratory study considered here,

the authors conducted a thematic analysis approach (Cruzes and Dyba, 2011; Thomas and Harden, 2008). Thematic analysis is a method for identifying, analyzing, and reporting patterns within the data. For that purpose, the data are chopped into small pieces of information, the quotations or segments, which are minimal units of data. Then, some individuals (typically some of the researchers) act as coders, codifying the segments to highlight the relevant information and assigning it a condensate description, the code. In the literature, codes are defined as "descriptive labels that are applied to segments of text from each study" (Cruzes and Dyba, 2011). To ease the task of the coders, the codes can be grouped into larger categories that share common higher-level characteristics, forming semantic domains (also known as themes in this context). This introduces a multi-level coding that usually leads to richer analysis.

A very important point is that the splitting of the matter under study into quotations can be provided by a non-coder individual (typically, the thematic analysis designer), or it can be a task delegated to the coders. In the former case, all coders work with the same segments, so it is easier to achieve a high level of consensus, which leads to high reliability in the results of the analysis. In the latter case, the coders can decide by themselves how to cut the stream of data, so hidden phenomena can be uncovered. However, the cuts may vary from one coder to another, so there exists a high risk of getting too diverse codings that cannot be analyzed under a common framework.

In the study considered in this section, the method followed for data analysis is described in the four phases described below (see also Fig. 2).

1. **Define quotations & codebook.** In the study under consideration, the coders used pre-defined quotations. In this way, once the interviews were transcripted, researcher R1 chopped the data into unit segments that remain unalterable during subsequent phases. In parallel, R1 elaborated a codebook by collecting all available codes and their aggregation into semantic domains. After completing the codebook, R1 also created a guide with detailed instructions on how to use the codebook and how to apply the codes.
   The design of the codebook is accomplished through two different approaches: a deductive approach (Miles and Huberman, 1994) and an inductive approach (Corbin and Strauss, 2008). The first phase was mainly addressed by a deductive approach, in which R1 created some semantic domains and codes from R1's previous knowledge. Thus, these initial domains and codes integrate concepts known in the literature. Each domain is named P01, P02, P03, etc. The codes were written in lowercase and identified by the domain name and consecutive letters e.g. the codes of the semantic domain P01 were named P01a, P01b, etc.
   The second phase was mainly addressed by an inductive approach, in which R1 approached the data (i.e. the interviews' transcriptions) with the research question RQ in mind. R1 reviewed the data line by line and created the quotations. R1 also assigned these quotations one or more codes (new or previously defined) in order to get a comprehensive list of all the needed codes. As more interviews were analyzed, the resulting codebook was refined using a constant comparison method that forced R1 to go back and forth. When necessary, R1 also created semantic domains in which codes are grouped inductively.
   Additionally, the codes were complemented with a brief explanation of the concept they describe. This allows R1 to guarantee that the collection of created codes satisfies the requirements imposed by thematic analysis, namely exhaustiveness and mutual exclusiveness. The exhaustiveness requirement means that the codes of the codebook must cover all relevant aspects of the research. Mutual exclusiveness means that there must be no overlap in the semantics of each code within a semantic domain. In this way, the codes of a particular semantic domain must capture disjoint aspects and complementary aspects, which implies that the codes should have explicit boundaries so that they are not interchangeable or redundant. This mutual exclusiveness translates into the fact that during the coding phase a coder cannot apply several codes of the same semantic domain to the same quotation. In other words, each coder can apply at most a code of each semantic domain to each quotation. This is an important semantic constraint aiming to disambiguate the use of codes. If, within a semantic domain, several codes share a common meaning, then the choice of which code must be applied to a quotation is arbitrary. This would lead to false disagreements.
2. **Code.** In this phase, the chosen coders (usually researchers different from the codebook designer) analyze the prescribed quotations created during phase (1). For that purpose, they use the codebook as a statement of the available semantic domains and codes, as well as the definitions of each one, scope of application, and boundaries. It is crucial for the process that the coders apply the codes exactly as described in the codebook. No modifications on the fly or alternative interpretations are acceptable.
   Nevertheless, the coders are encouraged to annotate any problem, diffuse limit, or misdefinition they find during the coding process. After the coding process ends, if the coders consider that the codebook was not clear enough or the ICA measured in phase (3) does not reach an acceptable level, the coders and the codebook designer can meet to discuss the found problems. With this information, the codebook designer creates a new codebook and coding instructions that can be used for a second round of coding. This iterative process can be conducted as many times as needed until the coders consider that the codebook is precise enough and the ICA measures certify an acceptable amount of reliability.
   In the case study of ICA considered in this paper, the coding process involved two researchers different from R1, who acted as coders C1 and C2. They coded the matter according to the codebook created by R1.
3. **Calculate ICA.** It is a quite common misconception in qualitative research that no numerical calculations can be performed for the study. Qualitative research aims to understand very complex and unstructured phenomena, for which a semantic analysis of the different facets and their variations is required. However, by no means, this implies that no mathematical measures can be obtained for controlling the process. Due to its broad and flexible nature, qualitative research is highly sensitive to introduce biases in the judgments of the researchers, so it is mandatory to supervise the research through some reliability measures that are usually numerical (Krippendorff, 2018). In this way, the quantitative approach takes place at a higher level, as a control of the conducted process in order to guarantee mathematical reliability in the drawn conclusions. Only when this formal quality assurance process is satisfactory, researchers can trust in the conclusions and the method is sound and complete.
   Therefore, to avoid biases and be confident that the codes mean the same to anyone who uses them, it is necessary to build that confidence. According to Krippendorff (2018),

reliability grounds this confidence empirically and offers the certainty that research findings can be reproduced.
In the presented example of a DevOps case study, we used Inter-Coder Agreement (ICA) analysis techniques for testing the reliability of the obtained codebook. In this way, after coding, another researcher, R4, calculated and interpreted the ICA between C1 and C2. If the coders did not reach an acceptable level of reliability, R1 analyzes the disagreements pointed out by R4 to find out why C1 and C2 had not understood a code in the same mode.
Using this acquired knowledge, R1 delivers a refined new version of the codebook and the accompanying use instructions. R1 also reviews the coding of those quotations that led to the disagreement between C1 and C2, modifying it according to the new codebook when necessary. Notice that, if a code disappears in the new version of the codebook, it also must disappear from all the quotations that were assigned with it.
At this point, C1 and C2 can continue coding on a new subset of interviews' transcriptions. This process was repeated until the ICA reached an acceptable level of reliability (typically $\geq$ 0.8). In Section 5.5 we provide a detailed explanation of how to compute and interpret ICA coefficients.

4. **Synthetize.** Once the loop (1)-(2)-(3) has been completed because the ICA measures reached an acceptable threshold, we can rely in the output of the coding process and start drawing conclusions. At this point, there exists a consensus about the meaning, applicability and limits of the codes and semantic domains of the codebook.
Using this processed information, this phase aims to provide a theory, model or taxonomy such that its constructs are built from the main semantic domains, and the relations between these constructs are inherited from the relations between the domains. To determine which semantic domains appear in the theory, two metrics are used: (i) how many times each domain was applied as a label in the analyzed data (grounded value), and (ii) the role of the domain in the relations, regarding both the number of times that this domain has been linked and the importance of the involved domains in the relation.
All these synthesis actions are not the main focus of this paper, so we will not describe them further. For more information and techniques, please refer to Wohlin et al. (2012).

5.4. Perform validation analysis

As a final step, it is necessary to discuss in which way the obtained analysis and drawn conclusions are valid, as well as the threats to the validity that may jeopardize the study. In the words of Wohlin et al. (2012) "The validity of a study denotes the trustworthiness of the results, and to what extent the results are true and not biased by the researchers' subjective point of view".

There are several strategies for approaching the validity analysis of the procedure. In the aforementioned case study, it was followed the methodology suggested by Creswell and Creswell (2017) to improve the validity of exploratory case studies, namely data triangulation, member checking, rich description, clarifying bias, and reporting discrepant information. Most of these methods are out of the scope of this paper and are not described further (for more information, check Creswell and Creswell, 2017; Wohlin et al., 2012). We mainly focus on reducing authors bias by evaluating the reliability and consistency of the codebook on which the study findings are based through ICA analysis.

5.5. ICA calculation

Let us further discuss the main concern of this paper: how to conduct the ICA analysis required to assess the validity of the exploratory study under review. For this purpose, we use the theoretical framework developed in Section 4 regarding the different variants of Krippendorff's α coefficient. In this way, we will monitor the evolution of the α coefficients throughout the coding process in order to assure it reaches an acceptable threshold of reliability, as mentioned in Section 5.3.

However, before starting the coding/evaluation protocol, it is worth considering two important methodological aspects, as described below.

1. The number of coders. Undoubtedly, the higher the number of coders involved, the richer the coding process. Krippendorff's α coefficients can be applied to an arbitrary number of coders, so there is no intrinsic limitation to this number. On the other hand, a high number of coders may introduce too many different interpretations that may make it difficult to reach an agreement. In this way, it is important to find a fair balance between the number of coders and the time to reach agreement. For that purpose, it may be useful to take into account the number of interviews to be analyzed, its length, and the resulting total amount of quotations. In the case study analyzed in this section, two coders, C1 and C2 were considered for coding excerpts extracted from 30 interviews.
2. The extend of the coding/evaluation loop. A first approach to the data analysis process would be to let the coders codify the whole corpus of interviews, and to get an resulting ICA measure when the coding is completed. However, if the obtained ICA is below the acceptable threshold (say 0.8), the only solution that can be given is to refine the codebook and to re-codify the whole corpus again. This is a slow and repetitive protocol that can lead to intrinsic deviations in the subsequent codings due to cognitive biases in the coders.
In this way, it is more convenient to follow an iterative approach that avoids these problems and speeds up the process. In this approach, the ICA coefficient is screened on several partially completed codings. To be precise, the designer of the case study splits the interviews into several subsets. The coders process the first subset and, after that, the ICA coefficients are computed. If this value is below the threshold of acceptance (0.8), there exists a disagreement between the coders when applying the codebook. At this point, the designer can use the partial coding carried out up to that moment in order to detect the problematic codes and to offer a refined version of the codebook and the accompanying instructions. Of course, after this revision, the previously coded matter should be updated with the new codes. With this new codebook, the coders can face the next subset of interviews, with the expectation that the newer version of the codebook will lead to a decrease in disagreement. This reduces the number of complete codings needed to achieve an acceptable agreement.
In the case study considered as example, the first batch of interviews comprised the first 19 interviews (ID01 to ID19). The attained ICA was unsatisfactory, so the codebook designer R1 reviewed the codebook releasing a new version. With the updated codebook, the coders codified the remaining 11 interviews (ID20 to ID30) but now the obtained ICA passes the acceptance threshold, which evidences a high level of reliability in the evaluations.

Table 10
Values of α_{binary} per semantic domain.

Semantic domain	α_{binary}	Semantic domain	α_{binary}
P01	1.0	P02	**0.651**
P03	1.0	P04	1.0
P05	1.0	P06	0.848
P07	0.913	P08	1.0
P09	0.872	P10	**0.796**

Table 11
Codified units, per document, for the semantic domain P07.

Code	Coder	Applied	Length quotations
P07a	A_1	2	388
	A_2	1	81
P07b	A_1	5	1143
	A_2	5	1143
P07c	A_1	2	403
	A_2	2	403

As a final remark, it is not recommendable to replace the coders during this iterative process. Despite that Krippendorff's α allows one to exchange coders, the new coders may not share the same vision and expertise with the codebook, requiring to roll back to previous versions (c.f. Remark 4).

Now, we present the calculation and interpretation of each of the four coefficients mentioned in Section 4. To emphasize the methodological aspects of the process, we focus only on some illustrative instances for each of the two rounds of the coding/evaluation protocol.

5.5.1. The α_{binary} coefficient

As mentioned in Section 4.2, the α_{binary} coefficients can be used to decide whether the semantic domains are clearly defined. In our case study, in the first round of coding we obtained the results shown in Table 10. In bold, we highlight those semantic domains that did not reach the acceptable threshold of reliability ($\geq$0.8). Hence, a refinement of the codebook was needed, and a second round of coding was conducted. In this second round, all the ICA coefficients reached the reliability threshold.

Notice that, for those semantic domains that reached a value of α_{binary} greater than the threshold of 0.8, it can be interpreted as evidence that the domain is clearly stated, its boundaries are well defined, and thus the decision of applying it or not is near to be deterministic. However, observe that this does not measure the degree of agreement in the application of the different codes within the domain. It might occur that the boundaries of the domain are clearly defined but the inner codes are not well chosen. This is not a task of the α_{binary}, but of the *cu-α* coefficient.

Typically, the computation of these α coefficients can be instrumented through an assistant software. However, to illustrate how these coefficients work, let us calculate α_{binary} by hand for the semantic domain P07. In Table 11 we show the part of the exported information that is relevant for our analysis. As we can see, there are two coders (A_1 and A_2) and three codes. The column "Applied" refers to the number of times the code has been applied, whereas "Length quotations" stands for the total length (expressed in the number of characters) of the quotations to which the code has been applied.

From this information, we see that coder A_1 voted 388 units (characters) with the first code of the domain (P07a) while coder A_2 only voted 81 units with that code. For the others codes, both coders apply them to 1143 and 403 units, respectively. Indeed, as we will check later, the quotations that A_2 chose for applying P07 are actually a subset of the ones chosen by A_1. Hence, A_1 and A_2 achieved perfect agreement when applying the second and third codes of P07 while A_2 only considered eligible 81 units for the first code of the 388 chosen by A_1.

Table 12
Observed coincidences matrix for α^{P07}_{binary}.

		Coder 2 (A_2)		
		1	0	
Coder 1 (A_1)	1	$o_{1,1} = 3254$	$o_{1,2} = 307$	$t_1 = 3561$
	0	$o_{1,2} = 307$	$o_{2,2} = 1004900$	$t_2 = 1005207$
		$t_1 = 3561$	$t_2 = 1005207$	$t = 1008768$

From these data, we can construct the observed coincidence matrix, shown in Table 12, as explained in Section 2.5 (see also Section 3). Recall from Section 4.2 that a label 1 means that the coder voted the quotation with a code of the semantic domain (P07 in this case) and the label 0 means that no code of the domain was applied.

This matrix is computed as follows. The number of units to which the coders assigned any code from domain P07 is $81 + 1143 + 403 = 1627$ in the case of A_2 and $388 + 1143 + 403 = 1934$ for A_1. Since the choices of A_2 are a subset of the ones of A_1, we get that they agreed in $1627 = \min(1627, 1934)$ units. Recall that $o_{1,1}$ counts ordered pairs of votes, so we need to double the contribution to get $o_{1,1} = 2 \cdot 1627 = 3254$. On the other hand, A_2 did not apply any code of P07 to $504384 - 1627 = 502757$ units, while A_1 did not apply them to $504384 - 1934 = 502450$, which means that they agreed on not to chose P07 in $502450 = \min(502757, 502450)$ units. Doubling the contribution, we get $o_{2,2} = 1004900$. Finally, for the disagreements we find that A_1 applied a code from P07 to 307 units that A_2 did not select, so we get that $o_{1,2} = o_{2,1} = 307$. Observe that we do not have to double this value, since there is already an implicit order in this votes (A_1 voted 1 and A_2 voted 0). From these data, it is straightforward to compute the aggregated quantities $t_1 = o_{1,1} + o_{1,2} = 3561$, $t_2 = o_{1,2} + o_{2,2} = 1005207$ and $t = t_1 + t_2 = 1008768$.

In the same vein, we can construct the matrix of expected coincidences, as explained in Section 2.5 (see also 3). The value of the expected disagreements are

$$e_{1,2} = e_{2,1} = \frac{t_1 t_2}{t - 1} = \frac{3561 \cdot 1005207}{1008767} = 3548.43.$$

Analogously, we can compute $e_{1,1}$ and $e_{2,2}$. However, they are not actually needed to calculate the α coefficient, so we will skip them. With these calculations, we finally get that

$$D_0 = o_{1,2} = 307, \quad D_e = e_{1,2} = 3548.43,$$
$$\alpha^{P07}_{binary} = 1 - \frac{D_o}{D_e} = 1 - \frac{307}{3548.43} = 0.913.$$

We want to again notice that the previous calculation is correct because A_2 voted with a code of P07 a subset of the quotations that A_1 selected for domain P07. To check this claim, Table 13 shows an excerpt of the displayed information. In this table, we see that all the voted elements coincide except the last 307 units corresponding to document ID17, that A_1 codified and A_2 did not.

It is worth mentioning that, in some situations, extremely low values of α_{binary} can be obtained that can mislead the researchers. Despite their undesirable appearance, they are nothing but a manifestation of a lack of variability in the data. To illustrate this phenomenon, we focus on the α_{binary} for P07 in the second round of the coding process, where a value $\alpha^{P07}_{binary} = -0.011$ was reported.

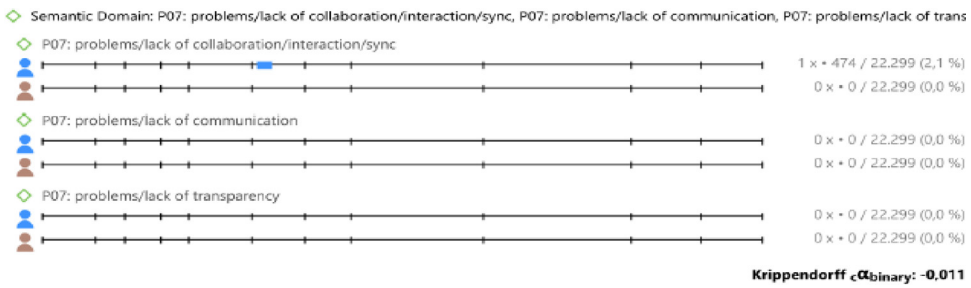

Fig. 3. Computation of the α_{binary} coefficient for the domain P07 in the second round.

Table 13
Codified units, per document, for the semantic domain P07.

Document ID	Coder 1 (A_1)	Coder 2 (A_2)
ID01	7b 1 × 112 & 7c 1 × 306	7b 1 × 112 & 7c 1 × 306
ID03	7b 1 × 185	7b 1 × 185
ID05	7b 1 × 159	7b 1 × 159
ID10	7a 1 × 81	7a 1 × 81
ID11	7b 1 × 314	7b 1 × 314
ID12	7b 1 × 373 & 7c 1 × 97	7b 1 × 373 & 7c 1 × 97
ID17	7a 1 × 307	

This is an extremely small value that might even point out a deliberate disagreement between the coders. However, this is not happening here, but an undesirable statistical effect that fakes the result. The point is that, as shown in Fig. 3, in round 2 there is only one evaluation from one of the coders that assigns this semantic domain, in contrast with the 17 evaluations obtained in round 1. For this reason, there are not enough data for evaluating this domain in round 2 and, thus, this result can be attributed to statistical outliers. Researchers interested in using ICA for qualitative research should be aware of this annoying phenomenon. In this case, due to the few received codings, the reliability may be assessed by hand.

Finally, as we mentioned in Section 4.1, the α^{gl}_{binary} coefficient allows researchers to measure the degree of agreement that the coders reached when distinguishing relevant and irrelevant matter. In this way, α^{gl}_{binary} is only useful if each coder chops the matter by him/herself to select the relevant information to code. On the other hand, if the codebook designer pre-defines the quotations to be evaluated, this coefficient is no longer useful since it always attains the value $\alpha^{gl}_{binary} = 1$. This latter situation is the case in our running example.

5.5.2. The cu-α and Cu-α coefficients

Recall from Section 4.3 that the binary coefficients α_{binary} and α^{gl}_{binary} evaluate whether to apply a particular semantic domain is well-defined in the codebook. They are binary measures, in the sense that they test a binary question: to apply some domain or not.

On the other hand, in this section we will consider the *cu-α* and *Cu-α* coefficients that, roughly speaking, zoom in to measure the limits of definition of the semantic domains themselves and of the codes within them. Recall from Section 4.3 that the *cu-α* coefficient is computed per semantic domain. Hence, fixed a domain S, it evaluates the amount of agreement that the coders reached when choosing to apply some code of S or other. Therefore, it is a measure of reliability in the application of codes within S, not of the domains themselves. Analogously, as explained in Section 4.4, *Cu-α* is a global measure that allows us to assess the limits of definitions of semantic domains. In other words, it measures the goodness of the partition of the codebook into semantic domains, independently of the chosen code.

Table 14
Values of *cu-α* per semantic domain.

Semantic domain	*cu-α*	Semantic domain	*cu-α*
P01	**0.705**	P02	1.0
P03	0.962	P04	1.0
P05	1.0	P06	**0.739**
P07	1.0	P08	1.0
P09	1.0	P10	**0.563**

In Table 14, we show the obtained results of *cu-α* for each of the 10 semantic domains of the running example considered in this section in the first round. The coefficients show that most of the semantic domain reached the acceptance threshold (≥ 0.8). This means that there exists a consensus on the meaning and applicability of the codes within these domains. However, a few of them (highlighted in bold) did not meet the standards. This points out that further analyses of their inner codes were needed since their description in the codebook were not totally clear and presented some type of ambiguity. For this reason, the review of the codebook also involved the revision of the codes of these domains.

As an example of this revision process for the domain P01, it was observed that both coders were using, in several quotations, the following two codes indistinguishably:

1. *"Need of being more agile, rapid: Need to reduce time to market, need of being more agile/rapid to adapt to market needs (demands) either new features or updates. Business demands more velocity than the velocity teams can offer. = Accelerate time to market. = Accelerate value delivery. = Faster time to market. = Rapid customer response".*
2. *"Too much time for releasing new features into production: Too much time for developing, testing, deploying, delivering, releasing new features and/or hotfixes. (If releasing takes too long, so you run the risk that the product is unsatisfactory)".*

After detecting this issue, we checked that the differences between these two codes were too subtle (putting your emphasis on the market for the first code or on the business/client for the second code), so we finally decided to unify both codes into a single one:

"Too much time for releasing new features into production: Need to reduce time to market, need of being more agile/rapid to adapt to market needs (demands) either new features or updates. Business demands more velocity than the velocity teams can offer. Too much time deploying, delivering, releasing new features and/or hotfixes. (If releasing takes too long, so you run the risk that the product is unsatisfactory). = Accelerate time to market. = Accelerate value delivery. = Faster time to market. = Rapid customer response".

In the second round of coding, all the domains achieved the acceptance threshold.

It is worth noticing that *cu-α* attained its maximum value, *cu-α* = 1, over the domains P02, P05, P07, P08, and P09. However, this may seem counter-intuitive at a first sight since, as we mentioned in Section 5.5.1, in P07 there is no perfect agreement. Indeed, as we know, the code P07a was chosen by A_1 for a quotation that A_2 skipped. This is strongly related to Remarks 1 and 9 since recall that, fixed a domain S, in the observed coincidences matrix only quotations were voted with codes of S by at least two different coders count. Otherwise, these quotations do not contribute with a pair of disagreements so, through the eyes of *cu-α*, they do not compromise the reliability.

This fact is precisely what is taking place in this case. The quotation voted by A_1 with a code of P07 and not by A_2 does not appear in the observed coincidences matrix for $cu\text{-}\alpha^{\text{P07}}$, neither as an agreement nor as a disagreement. This allows $cu\text{-}\alpha^{\text{P07}} = 1$ even though there is no perfect agreement in the evaluation. This might seem awkward, but it actually makes sense, since this disagreement was already detected via $\alpha^{\text{P07}}_{binary} < 1$, so decreasing also $cu\text{-}\alpha^{\text{P07}}$ would count it twice. The same scenario occurs in domains P02 and P09.

Finally, the *Cu-α* coefficient reached a value of 0.67, which is slightly above the lower threshold of applicability of 0.667. This suggests that the limits of definition of the semantic domains are a bit diffuse and can be improved. This problem was addressed in the second version of the codebook, in which a better definition of the domains allowed us to increase *Cu-α* to 0.905, which is a sound evidence of reliability.

6. Threats and limitations

The theoretical framework introduced in this work presents no limitations related to the experimental setup. This framework enables the use of the four variants of the Krippendorff's α coefficients for any type of data (nominal, ordinal, interval or ratio), allows missing codes, takes into account the agreement by chance, and provides thresholds to interpret their outputs. It is also worth noticing that these coefficients work with arbitrarily many coders, even if they act anonymously, but we encourage finding a trade-off between the number of coders (more coders speeds up the coding process) and the possibilities of reaching a consensus quickly (the more coders, the harder the agreement).

On the other hand, it must be highlighted that the use of this methodology imposes an intrinsic limitation in the design of the codebook. As mentioned in Section 4, "the semantic domains must be logically and conceptually independent. This principle translates into the fact that there exist no shared codes between different semantic domains and two codes within the same semantic domain cannot be applied at the same time by a coder to the same quotation". These limitations are imposed by the requirements of thematic analysis, namely exhaustiveness and mutual exclusiveness. The exhaustiveness requirement means that the codes of the codebook must cover all relevant aspects of the research. Mutual exclusiveness means that there must be no overlap in the semantics of each code within a semantic domain.

Regarding the threats, in this research we have reported some behaviors of the coefficients that might seem 'erratic', such as those due to a lack of variability in the data (Section 5.5.1). These values can mislead the researchers involved in the study, who can erroneously believe that they correspond to incorrect values. To address this issue, in this work, we have pointed out the underlying causes of these behaviors, with a view towards helping researchers to interpret them in the right way. We have also provided procedures to bypass these difficulties so that reliability may be assessed by hand in these cases.

7. Conclusions

Throughout this work, we have discussed a set of statistical measures proposed in the literature for evaluating Inter-Coder Agreement in thematic analysis. Among them, we have paid special attention to Krippendorff's α coefficients as the most appropriate and best behaved for qualitative analysis.

Additionally, we have introduced a formal setting in which these coefficients can be formalized. We have presented a novel theoretical framework in which we provide a common formulation for all of them in terms of a universal α coefficient. This analysis provides a clearer and more precise interpretation of four of the most important variants of the α coefficients: the binary α coefficient (α_{binary}), the global binary α coefficient (α^{gl}_{binary}), the cu coefficient (*cu-α*) and the Cu coefficient (*Cu-α*). This redefinition is particularly well suited for its use in case studies, interview surveys, and grounded theory, with a view towards providing sound reliability of the coding.

In this paper, we have presented a running example based on an exploratory study about the adoption of the DevOps culture in software companies in order to illustrate how to apply these coefficients to software engineering research to improve the reliability of the drawn conclusions. With this idea in mind, we described how to compute and interpret Krippendorff's α coefficients.

Furthermore, the interpretation provided in this paper has allowed us to detect some bizarre behaviors of the Krippendorff's α coefficients that are not typically described in qualitative research manuals and that may mislead researchers who are not familiar with the α measures. Shedding light upon these unexpected results, justifying why they appear, and how to interpret them have been the guiding lines of this work. In particular, we addressed situations in which insufficient statistical variability leads to paradoxical results in which very small deviations from the agreement lead to very bad measures of the α coefficient and we clarified why *cu-α* may be maximum even though there is no perfect agreement and how to detect it through α_{binary}.

Most qualitative work in software engineering suffers from a lack of formal measures of the reliability of the conclusions drawn. This is a very dangerous threat that must be addressed to establish sound, well-posed, and long-lasting knowledge. Otherwise, if the data are not reliable, the conclusions drawn are not trustworthy. In this direction, we expect that this paper will help researchers in qualitative analysis, in general, and in empirical software engineering, in particular, incorporate these techniques into their investigations, aiming to improve the quality and reliability of the research.

Finally, we are confident (as it is part of the scientific community) that it is possible to combine quantitative research techniques with qualitative research. We agree with Braun & Clarke's statement: "ICA does not necessarily imply that there is a single true meaning inherent in the data, which is the concern underlying most epistemological objections, but shows that a group of researchers working within a common conceptual framework can arrive at a consensual interpretation of the data" (Braun and Clarke, 2013). In line with the words of M. Patton (Patton, 1999), the debate between qualitative and quantitative methodologists is often strident: "The important challenge is to match appropriately the methods to empirical questions and issues, and not to universally advocate any single methodological approach for all problems".

CRediT authorship contribution statement

Ángel González-Prieto: Conceptualization, Methodology, Formal analysis, Investigation Writing – original draft, Writing –

review & editing, Supervision. **Jorge Perez:** Conceptualization, Methodology, Formal analysis, Investigation, Writing – original draft, Writing — review & editing, Supervision. **Jessica Diaz:** Methodology, Software, Validation, Data curation, Writing – original draft, Writing — review & editing. **Daniel López-Fernández:** Methodology, Resources, Validation, Data curation, Writing – original draft, Writing — review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Reference to the original publication with a link to the repository with the used data is provided in the manuscript

Acknowledgments

The first author has been partially supported by the Madrid Government (Comunidad de Madrid – Spain) under the Multi-annual Agreement with the Universidad Complutense de Madrid in the line Research Incentive for Young PhDs, in the context of the V PRICIT (Regional Programme of Research and Techno logical Innovation) through the project PR27/21-029 and by the Ministerio de Ciencia e Innovación Project, Spain PID2021-124440NB-I00 (Spain).

References

ATLAS.ti Scientific Software Development GmbH, 2020. Inter-coder agreement analysis: Atlas.ti 9. URL https://atlasti.com/.

Berntzen, M., Hoda, R., Moe, N.B., Stray, V., 2023. A taxonomy of inter-team coordination mechanisms in large-scale agile. IEEE Trans. Softw. Eng. 49 (2), 699–718. http://dx.doi.org/10.1109/TSE.2022.3160873.

Bhasin, T., Murray, A., Storey, M.-A., 2021. Student experiences with GitHub and stack overflow: An exploratory study. In: 2021 IEEE/ACM 13th International Workshop on Cooperative and Human Aspects of Software Engineering. CHASE, pp. 81–90. http://dx.doi.org/10.1109/CHASE52884.2021.00017.

Braun, V., Clarke, V., 2013. Successful Qualitative Research: A Practical Guide for Beginners. sage.

Cicchetti, D.V., Feinstein, A.R., 1990. High agreement but low kappa: II. Resolving the paradoxes. J. Clin. Epidemiol. 43 (6), 551–558.

Cohen, J., 1960. A coefficient of agreement for nominal scales. Educ. Psychol. Meas. 20 (1), 37–46.

Corbin, J., Strauss, A., 2008. Techniques and procedures for developing grounded theory. In: Basics of Qualitative Research, third ed. Sage, Thousand Oaks, CA, USA.

Craggs, R., Wood, M.M., 2005. Evaluating discourse and dialogue coding schemes. Comput. Linguist. 31 (3), 289–296.

Creswell, J.W., Creswell, J.D., 2017. Research Design: Qualitative, Quantitative, and Mixed Methods Approaches. Sage publications.

Cronbach, L.J., 1951. Coefficient alpha and the internal structure of tests. Psychometrika 16 (3), 297–334.

Cruzes, D.S., Dyba, T., 2011. Recommended steps for thematic synthesis in software engineering. In: 2011 International Symposium on Empirical Software Engineering and Measurement. IEEE, pp. 275–284.

De Swert, K., 2012. Calculating inter-coder reliability in media content analysis using Krippendorff's Alpha. Cent. Politics Commun. 1–15.

Díaz, J., López-Fernández, D., Pérez, J., González-Prieto, Á., 2021. Why are many businesses instilling a DevOps culture into their organization? Empir. Softw. Eng. 26 (2), 1–50.

Díaz, J., Pérez, J., Gallardo, C., González-Prieto, Á., 2023. Applying inter-rater reliability and agreement in collaborative grounded theory studies in software engineering. J. Syst. Softw. (ISSN: 0164-1212) 195, 111520. http://dx.doi.org/10.1016/j.jss.2022.111520, URL https://www.sciencedirect.com/science/article/pii/S0164121222001960.

Feinstein, A.R., Cicchetti, D.V., 1990. High agreement but low kappa: I. The problems of two paradoxes. J. Clin. Epidemiol. 43 (6), 543–549.

Feng, G.C., 2014. Intercoder reliability indices: disuse, misuse, and abuse. Qual. Quant. 48 (3), 1803–1815.

Feng, G.C., 2015. Mistakes and how to avoid mistakes in using intercoder reliability indices. Methodol. Eur. J. Res. Methods Behav. Soc. Sci. 11 (1), 13.

Fleiss, J.L., 1971. Measuring nominal scale agreement among many raters. Psychol. Bull. 76 (5), 378.

Friese, S., 2019. Qualitative Data Analysis with ATLAS.ti. SAGE Publications Limited.

Ghanbari, H., Vartiainen, T., Siponen, M., 2018. Omission of quality software development practices: A systematic literature review. ACM Comput. Surv. (ISSN: 0360-0300) 51 (2), http://dx.doi.org/10.1145/3177746.

Gisev, N., Bell, J.S., Chen, T.F., 2013. Interrater agreement and interrater reliability: key concepts, approaches, and applications. Res. Soc. Adm. Pharm. 9 (3), 330–338.

Gwet, K.L., 2011. On the Krippendorff's alpha coefficient. https://agreestat.com/papers/onkrippendorffalpha_rev10052015.pdf, Preprint.

Hayes, A.F., Krippendorff, K., 2007. Answering the call for a standard reliability measure for coding data. Commun. Methods Meas. 1 (1), 77–89.

Hoda, R., 2022. Socio-technical grounded theory for software engineering. IEEE Trans. Softw. Eng. 48 (10), 3808–3832. http://dx.doi.org/10.1109/TSE.2021.3106280.

Holsti, O.R., 1969. Content Analysis for the Social Sciences and Humanities. Addison-Wesley (Content Analysis), Reading. MA.

Krippendorff, K., 1970. Estimating the reliability, systematic error and random error of interval data. Educ. Psychol. Meas. 30 (1), 61–70.

Krippendorff, K., 1995. On the reliability of unitizing continuous data. Sociol. Methodol. 47–76.

Krippendorff, K., 2018. Content Analysis: An Introduction To Its Methodology, fourth ed. Sage publications.

Krippendorff, K., Mathet, Y., Bouvry, S., Widlöcher, A., 2016. On the reliability of unitizing textual continua: Further developments. Qual. Quant. Int. J. Methodol. 50 (6), 2347–2364. http://dx.doi.org/10.1007/s11135-015-0266-1, URL https://ideas.repec.org/a/spr/qualqt/v50y2016i6d10.1007_s11135-015-0266-1.html.

Landis, J.R., Koch, G.G., 1977. The measurement of observer agreement for categorical data. Biometrics 33 (1), 159–174, URL http://www.jstor.org/stable/2529310.

Lantz, C.A., Nebenzahl, E., 1996. Behavior and interpretation of the κ statistic: Resolution of the two paradoxes. J. Clin. Epidemiol. 49 (4), 431–434.

Leite, L., Rocha, C., Kon, F., Milojicic, D., Meirelles, P., 2019. A survey of DevOps concepts and challenges. ACM Comput. Surv. (ISSN: 0360-0300) 52 (6), http://dx.doi.org/10.1145/3359981.

Madampe, K., Hoda, R., Grundy, J., 2023. The emotional Roller Coaster of responding to requirements changes in software engineering. IEEE Trans. Softw. Eng. 49 (3), 1171–1187. http://dx.doi.org/10.1109/TSE.2022.3172925.

Miles, M.B., Huberman, A.M., 1994. Qualitative Data Analysis: An Expanded Sourcebook. sage.

Nili, A., Tate, M., Barros, A., 2017a. A critical analysis of inter-coder reliability methods in information systems research. In: Proceedings of the 28th Australasian Conference on Information Systems. University of Tasmania, pp. 1–11.

Nili, A., Tate, M., Barros, A., Johnstone, D., 2020. An approach for selecting and using a method of inter-coder reliability in information management research. Int. J. Inf. Manage. (ISSN: 0268-4012) 54, 102154. http://dx.doi.org/10.1016/j.ijinfomgt.2020.102154, URL http://www.sciencedirect.com/science/article/pii/S0268401219308564.

Nili, A., Tate, M., Johnstone, D., 2017b. A framework and approach for analysis of focus group data in information systems research. Commun. Assoc. Inf. Syst. 40.

Patton, M.Q., 1999. Enhancing the quality and credibility of qualitative analysis. Health Serv. Res. 34 (5 Pt 2), 1189.

Pearson, K., 1895. VII. Note on regression and inheritance in the case of two parents. Proc. R. Soc. Lond. 58 (347–352), 240–242.

Pereira, C., Santos, A., Machado, L., Zaina, L., 2022. How developers feel about tools: an investigation on software startup professionals experience with virtual kanban boards. In: 2022 IEEE/ACM 15th International Workshop on Cooperative and Human Aspects of Software Engineering. CHASE, pp. 1–10. http://dx.doi.org/10.1145/3528579.3529172.

Pérez, J., Díaz, J., Garcia-Martin, J., Tabuenca, B., 2020. Systematic literature reviews in software engineering-enhancement of the study selection process using Cohen's Kappa statistic. J. Syst. Softw. 110657.

Rumbaugh, J., Jacobson, I., Booch, G., 1999. The unified modeling language. In: Reference Manual. Addison Wesley Longman Inc.

Saldaña, J., 2012. The Coding Manual for Qualitative Researchers. Sage publications.

Salleh, N., Hoda, R., Su, M.T., Kanij, T., Grundy, J., 2018. Recruitment, engagement and feedback in empirical software engineering studies in industrial contexts. Inf. Softw. Technol. (ISSN: 0950-5849) 98, 161–172. http://dx.doi.org/10.1016/j.infsof.2017.12.001, URL http://www.sciencedirect.com/science/article/pii/S0950584917303786.

Scott, W.A., 1955. Reliability of content analysis: The case of nominal scale coding. Public Opin. Q. 321–325.

Spearman, C., 1904. The proof and measurement of association between two things. Am. J. Psychol. (ISSN: 00029556) 15 (1), 72–101, URL http://www.jstor.org/stable/1412159.

Stol, K.-J., Ralph, P., Fitzgerald, B., 2016. Grounded theory in software engineering research: A critical review and guidelines. In: Proceedings of the 38th International Conference on Software Engineering. ICSE '16, Association for Computing Machinery, New York, NY, USA, ISBN: 9781450339001, pp. 120–131. http://dx.doi.org/10.1145/2884781.2884833.

Storer, T., 2017. Bridging the chasm: A survey of software engineering practice in scientific programming. ACM Comput. Surv. (ISSN: 0360-0300) 50 (4), http://dx.doi.org/10.1145/3084225.

Thomas, J., Harden, A., 2008. Methods for the thematic synthesis of qualitative research in systematic reviews. BMC Med. Res. Methodol. 8 (1), 45.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A., 2012. Experimentation in Software Engineering. Springer Science & Business Media.

Yang, Y., Green, S.B., 2011. Coefficient alpha: A reliability coefficient for the 21st century? J. Psychoeduc. Assess. 29 (4), 377–392.

Zhao, X., Liu, J.S., Deng, K., 2013. Assumptions behind intercoder reliability indices. Ann. Int. Commun. Assoc. 36 (1), 419–480.

Ángel González-Prieto received a Double BSc. degree in Computer science and Mathematics from Universidad Autónoma de Madrid in 2014, a MSc. in Mathematics from Universidad Autónoma de Madrid in 2015, and a Ph.D. in Mathematics from Universidad Complutense de Madrid (Extraordinary Award), in 2018. Since then, he has been assistant professor at Universidad Autónoma de Madrid, postdoc at the Instituto de Ciencias Matemáticas (CSIC), Spain, and research assistant at the Universidad Politécnica de Madrid, Spain. Currently, he is assistant professor at the Faculty of Mathematical Sciences at Universidad Complutense de Madrid, Vicedean for International and Institutional Affairs at the same institution and member of the Instituto de Ciencias Matemáticas. His research interests include theoretical and applied machine learning, statistical methods in software engineering, algebraic geometry, and algebraic topology.

Jorge Perez received the BS degree in computer science from the Universidad Carlos III de Madrid, Spain, in 1999, and the Ph.D. degree from the Universidad Politécnica de Madrid (UPM), Spain, in 2004. From 2007 to 2014, he was the headmaster of the Department of Applied Computer Science, UPM, Spain, where he has been an associate professor, since 1985. He is currently the coordinator of the DMAE-DIA Educational Innovation Group. He has taken part in the creation of several syllabuses for both grades and Postgraduates and has supervised many research projects. His research interests include software architectures and meta-modeling, and qualitative research in software engineering. He is a recipient of several awards from the Rector of the expand for educational innovation at the university.

Jessica Diaz received the Ph.D. degree in computer science from UPM, Spain, in 2012 (best thesis Award). She is an associate professor at the Universidad Politécnica de Madrid (UPM, E.T.S. Ingeniería de Sistemas Informáticos), Spain. Since April 2003, she is a researcher in System and Software Technology Research Group (SYST) and is participating in several european and national projects related to Software Engineering on Internet of Things and Smart Systems. Her research interests are focused on empirical software engineering, research methods in SE, agile software development, DevOps, IoT, cloud computing, model-driven development, and software product lines. Currently she is researching about DevOps culture and practice in industry and contributing in the ACM SIGSOFT Empirical Standards.

Daniel López-Fernández received the graduate degree in software engineer and the Ph.D. degree in software & systems from the Technical University of Madrid (UPM), Spain. Regarding his experience, it is worth noting that he has taught agile methods in several universities and has researched and provided training about agile methods in multinational companies. Currently, he works as associate professor at UPM, Spain. His main research interests include the study and application of agile methodologies and DevOps culture and practices in professional environments, as well as the application of active learning methods and the development of innovative educational tools such as Serious Video Games, Escape Room and Virtual Reality applications.