Recibido 31 Oct. 2023 *ReCIBE, Año 12 No. 2, Nov. 2023*

Aceptado 01 Nov. 2023

Software testing using the code review technique: an exploratorystudy

Pruebas de software utilizando la técnica de revisión de código: un estudio exploratorio

Juan Pablo Ucán Pech[1]
Raúl Antonio Aguilar Vera [1*]
avera@correo.uady.mx
Julio César Díaz Mendoza[1]
Antonio Armando Aguileta Güemez[1]

[1] *Facultad de Matemáticas,Universidad Autónoma de Yucatán,Mérida, Yucatán, México.*
*Corresponding Author

ABSTRACT

This paper presents an exploratory study with Belbin roles, in particular role types, in an individual activity consisting of code review. The objective is to identify if, in addition to any of the three types of roles, the position of the fault and the gender of the subject influence the activity of detecting faults in the code. To create an experimental context for the review process, during a work session the subjects, who were software engineering students, used a code with injected faults for their review. With respect to the types of roles, the results of the experiment do not show significant differences either in the efficiency index obtained by the subjects in the testing process, or with the confusion index of failures. On the other hand, regarding the position of the fault, the results show significant differences between the faults detected in the first half of the code with respect to the remaining second half. Regarding the gender of the subject, the experiment does not show a significant difference between the detected faults.
At the end of this report, researchers should perform a short-term analysis of the faults introduced in the code to obtain a better version of the experimental object, allowing them to perform a second controlled experiment under less restrictive conditions.

KEYWORDS

Code Review, Experimentation, Faults in the code, Software Testing.

RESUMEN

Este artículo presenta un estudio exploratorio con los roles de Belbin, en particular los tipos de roles en una actividad individual que consiste en la revisión de código. El objetivo es identificar si, además de alguno de los tres tipos de roles, la posición de la falta y el género del sujeto influyen en la actividad de detección de faltas en el código. Para crear un contexto experimental para el proceso de revisión, durante una sesión de trabajo los sujetos, que eran estudiantes de ingeniería de software, utilizaron un código con faltas inyectadas para su revisión. Respecto a los tipos de roles, los resultados del experimento no muestran diferencias significativas en el índice de eficiencia obtenido por los sujetos en el proceso de prueba, así como tampoco en el índice de confusión de fracasos. Por otro lado, en cuanto a la posición de la falta, los resultados muestran diferencias significativas entre las faltas detectadas en la primera mitad del código respecto al resto de la segunda mitad del código. En cuanto al género del sujeto, el experimento no muestra una diferencia significativa entre las faltas detectadas.
Al final de este reporte, los investigadores deberán realizar un análisis a corto plazo de las faltas introducidas en el código para obtener una mejor versión del objeto experimental, lo que les permitirá realizar un segundo experimento controlado en condiciones menos restrictivas.

PALABRAS CLAVE

Revisión de Código, Experimentación, Faltas en el código, Pruebas de Software.

1. INTRODUCTION

The body of knowledge developed and accumulated over half a century after the so-called Software Crisis (Bourque & Fairley, 2014) has maintained a dynamic of constant improvement in terms of quality, both of its processes and of the artifacts that are generated by the first. The studies reported on software development and management processes have been analyzed based on various factors; however, the intrinsic social aspect of the discipline (Juristo & Moreno, 2001) has led to consider the human factor as an aspect of unique importance to your research. (Morales & Vega, 2018) proposes a catalog of human factors that are critical to the success of proposals to improve the software process, and among these factors is the role played by a Software Engineer within the work team.

In studies on the roles played by team members, a distinction is made between roles that focus on individual activities, on the one hand, and roles that are described in terms of the tasks they can perform as a team. Among studies on roles, Belbin's propositions (Belbin, 1981, 1993) are among the most popular in academic and professional contexts.

The purpose of this research is to expand the study related to the use of Belbin roles, in particular the types of roles, in an individual activity—in the context of software development—that corresponds to code review. The aim is to identify whether, in addition to any of the three types of roles, the position of the fault and the gender of the subject have an influence on the fault detection activity.

The following section presents the theoretical framework that supports Belbin's role theory, as well as the fault detection. Section three, four and five presents three analysis a controlled experiment carried out with teams of students – as experimental subjects – in the software testing task, particularly static code review. The three analysis was by role type, fault position and by gender respectively. Finally, section six presents the conclusions of the empirical study, as well as future work identified by the researchers.

2. BACKGROUND

2.1. Belbin Roles

Belbin (1981, 1993) maintains that a team role refers to the way of behaving, contributing and relating to other people at work and although some of the roles are natural, other roles could be adopted by the individual himself and some may even be discovered after being adopted. The nine roles proposed by Belbin can be grouped around the type of conduct in three different categories as described in Table 1.

Table 1. Belbin Roles Categories

Type	Rol	Characteristics
Action	Sharper (S) Implementer (I) Completer-Finisher (CF)	They are those roles that initiate, develop and finish tasks.
Mental	Plant (P) Monitor-Evaluator (ME) Specialist (SP)	They are those roles that have the knowledge and skills required for the task, as well as a critical vision for its realization.
Social	Chairman (CH) Resource Investigator (RI) Teamworker (TW)	They are those that promote communication and cohesion both among the members of the team and with the people with whom the team interacts.

Estrada & Peña (2013) presents the use of Belbin's role theory in individual tasks in the context of Software Engineering. This reported a controlled experiment with students performing activities related to the stages of requirements, design and coding; the authors conclude that some roles have greater input in certain activities, particularly pointing to the Implementer role in the coding task.

Another study is reported in (Aguilar et al., 2022), in which the possible differences between the nine roles in tasks related to the Logical Design of a Database are explored; the authors report that the Monitor-Evaluator role presents significant differences, having obtained a better quality grade than the other six roles participating in the experiment.

2.2. Fault Detection In The Code

The way that Software Engineering has to evaluate a software artifact is known as Empirical Verification, this way of acting does not provide a definitive solution.

When we evaluate code, on many occasions the words error and fault are often confused or used without distinction as if they were the same. In a similar way it occurs with two other words such as failure and defect, however, they are not the same. For example in the Figure 1 we can see that one system failure occurs when an output of a program it does not match with the established requirements, but a system failure it may be a fault that occurs as results of errors. On the other hand, the results of errors are when modeling, code or other artifacts do not comply with established requirements.

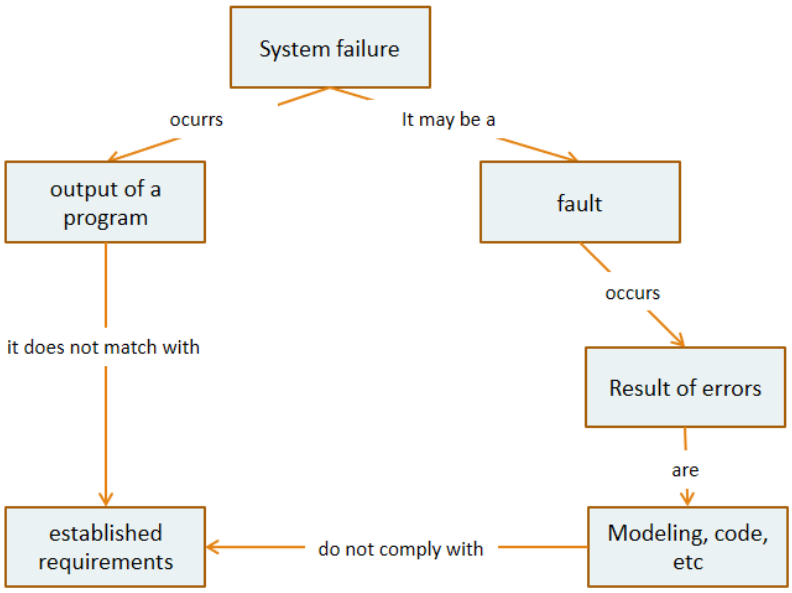

Figure 1. Failure, fault, error and their relations.

According to Juristo et al. (2006), there are four concepts that are usually used indifferently, and however they represent different constructs:

- Error: human action that produces a fault.
- Fault: something is wrong with a product (model, code, document, etc.).
- Failure: manifestation of a fault.
- Defect: error, fault or failure.

Classifying coding faults is not an easy task, in the literature; it is possible to find a variety of classifications about types and causes of faults that can be made, mainly by novice programmers. These classifications have been generated as a result of empirical studies on patterns found in programming activity (Ucán et al., 2022). Basili & Selbi (1987) exposes a typology indicating that faults can be classified in the following six classes of faults: Initialization, Computation, Control, Interface, Data and Cosmetic.

3. EXPERIMENTAL ANALYSIS I: BY ROLE TYPE

The purpose of this controlled experiment is to explore whether any of the types of roles proposed by Belbin (Action, Mental or Social) present a performance in the code review process that is statistically different from that of the other two. Two performance indicators have been particularly considered: Effectiveness and Confusion.

Effectiveness Index: The quotient of the number of faults correctly detected by the experimental subject, between the total number of faults injected.

Confusion Index: The quotient of the total number of incorrect errors detected between the total number of errors reported by the experimental subject.

The controlled experiment was carried out with students who completed their first programming course. According to the classification of programming experience created by Dreyfus & Dreyfus (1986), the experimental subjects can be classified as beginning students.

3.1. Factor and alternatives

According to Table 1, Belbin classified the nine team roles into three types of roles: Action Roles (AR), Mental Roles (MR), and Social Roles (SR); therefore, in this first experimental analysis, the type of role is considered as a factor, and the three types as alternatives: AR, MR and SR.

3.2. Hypotesis and variables

The first pair of statistical hypotheses uses as a dependent variable, a metric linked to the quality of the individual process, the effectiveness (effectiveness index) in identifying faults in the code.

- H_{01}: The means of the effectiveness index in the detection of faults in the code, by the three Types of Belbin Roles, do not present differences.
- H_{11}: The means of the effectiveness index in the detection of faults in the code, by the Belbin Role Types, differ in at least a couple of these.

The second variable is linked to the error made by the evaluator in the fault identification task, when incorrectly identifying a type of fault; human confusion (confusion index) will be used as a metric in the identification of faults in the code. The statistical hypotheses derived from said variable are the following:

- H_{02}: The means of the confusion index in the detection of faults in the code, by the three Types of Belbin Roles, do not present differences.
- H_{12}: The means of the confusion index for the detection of faults in the code, by the Belbin Role Types, differ in at least a couple of these.

3.3. Experimental unit

The experimental unit, also known as the experimental object, is the piece or sample that is used to generate a value that is representative of the result of the experiment. In our study, since the activity is the detection of faults in the code, the experimental object is the code reviewed by the subjects during the experiment.

Date: _____

Name: _____

Begin time: _____

End time: _____

Number of faults found: _____

Faults descriptions

No.	Type	Line or position	Fault description

Figure 2 – Data collection instrument**.**

For the purposes of our study, twelve faults were injected into the code, these faults used the classification proposed in (Basili & Perricone, 1984) as a reference and were distributed throughout the 214 lines of code (LOC). For the activity to be carried out with the code, an instrument was designed in which the registration of the information required for the identification of the subject is requested, as well as each of the faults detected (see Figure 2).

3.4. Experimental design

The most appropriate experimental design for our study is the factorial design with a source of variation and three alternatives (see Table 2). The dependent variables are numerical metrics that are obtained from the analysis of the information obtained with the instrument illustrated in Figure 2, which will be used by the experimental subjects to record the information described in the previous section.

Table 2 –Factorial design with one source of variation and three alternatives

Factor	Alternatives	Dependent variables
Role type	Action Roles (Type 1)	Effectiveness, Confusion
	Social Roles (Type 2)	
	Mental Roles (Type 3)	

3.5. Experimental subjects

The convenience sample used in the experiment consisted of 24 of the students enrolled in the aforementioned course, these participants already had the essential knowledge of the C programming language as well as programming logic under the structured paradigm. To identify the type of role of each subject, at the end of one of the class sessions during the course, the Belbin self-perception test was applied to all students enrolled in the course. With this, the authors identified the role assumed — and therefore the type of role—for each of the possible subjects. Finally, the list was refined with the students who voluntarily wanted to collaborate with the study and participated in the experimental session. Genero et al., (2014) argued that a student-based sample allows the researcher to obtain preliminary evidence to confirm or refute hypotheses that can then be tested in an industrial context.

3.6. Descriptive Analisys

Table 3 presents some of the most important measures of central tendency and variability for the Effectiveness variable. We can identify that the mean and median are higher with type roles 1, although the type of role with the least variability is 2 also; it is also possible to observe that the unbalanced sample contains fewer subjects with types 2 and 3.

Table 3 –Statistical summary for the Efficiency Index

Type	#	Mean	Median	SD
1	15	0.410889	0.416667	0.23279
2	5	0.333333	0.333333	0.102062
3	4	0.3125	0.291667	0.142319
Total	24	0.378333	0.333333	0.198304

To compare the three alternatives, we generate a box-and-whisker plot. Such a plot can allow us to observe the dispersion and symmetry of the three data sets; in Figure 3 we can see less dispersion in types 2 and 3, however, there is no gap between the three data sets, which suggests that there is no significant difference between the three types of roles.

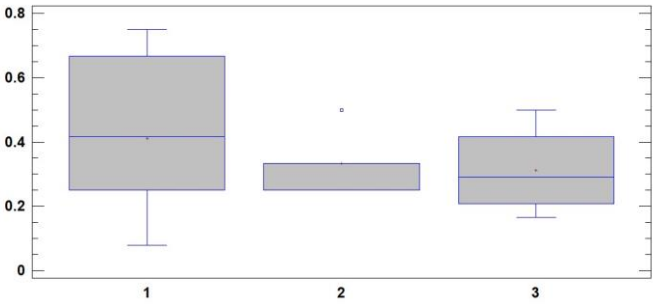

Figure 3 – Box and Whisker Plot for the Effectiveness.

Regarding the Confusion variable, Table 4 contains some of the most important measures. We can observe that type 1 presents a lower value in the mean and median, although there is a greater dispersion among the three types of roles.

Table 4 – Statistical summary for the Confusion Index

Type	#	Mean	Median	SD
1	15	0.405091	0.4	0.249227
2	5	0.512273	0.5	0.115615
3	4	0.5	0.5	0.148054
Total	23	0.443239	0.422619	0.213396

In Figure 4, we can observe and compare the mean, mode and dispersion of the three data sets and it is noteworthy that the roles of type 2 and 3 in 75% of the cases they coincide with 50% of those of type 1. That is, there is a certain gap, which will have to wait for the inferential analysis to determine if these differences are significant.

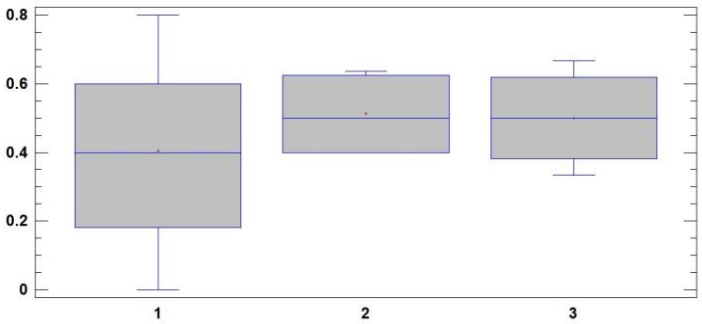

Figure 4 – Box and Whisker Plot for the Variable Confusion.

3.7. Inferencial analisys

With the purpose of statistically evaluating the differences between the alternatives of the Effectiveness and Confusion ratio variables, the one-way Analysis of Variance was applied (Gutierrez & De la Vara, 2012); the result of evaluating with ANOVA is illustrated in Table 5.

Table 5 –ANOVA for the Variables Effectiveness and Confusion

Metric	Ratio F	p-value
Effectiveness	0.53	0.5970
Confusion	0.62	0.5466

In both variables, a p-value much greater than 0.05 is obtained, which indicates that the null hypothesis is accepted; however, it is necessary to evaluate the model in both cases to be able to consider said hypotheses as true.

3.8. Validation of Model Assumptions

To validate the first assumption, Shapiro Wilk test allows us to evaluate whether a variable has a normal distribution or not (see Table 6).

Table 6 – Shapiro Wilk Test for the Variables Effectiveness and Confusion

Metric	Test	p-value
Effectiveness	0.945844	0.2198
Confusion	0.982342	0.9348

The p value for the Shapiro Wilk Test in both variables is greater than 0.05, therefore, it is possible to assume that both samples have a normal distribution. Regarding the second assumption, that of homoscedasticity, Levene's test allows the evaluation of significant differences between the variances of the two data sets (see Table 7).

Table 7 – Levene's Test for the Variables Effectiveness and Confusion

Metric	Test	p-value
Effectiveness	3.42508	0.0516
Confusion	1.91691	0.1719

The p value for the Levene test in both variables is greater than 0.05, which indicates that there is no statistically significant difference between the standard deviations of the alternatives, with a 95.0% confidence level; the above, both for the Effectiveness variable and for the Confusion variable.

Finally, for the assumption of data independence, although the subjects present independence as independent individuals, we chose to run the Durbin-Watson test, in order to identify if there is no relationship in at least the temporal sequence of the data. In the Table 8 the p value for the Durbin-Watson Test in both variables is greater than 0.05, which confirms our suspicion of independence.

Table 8 – Durbin-Watson Test for the Variables Effectiveness and Confusion

Metric	Test	p-value
Effectiveness	1.57585	0.1544
Confusion	2.06244	0.5553

4. EXPERIMENTAL ANALYSIS II: BY FAULT POSITION

The purpose of this second analysis is to explore whether the position of the fault has any influence on its detection. The correct number of faults detected in both halves of the code has been considered as a metric. However, because the faults are of different types, we have considered the metric on an ordinal scale.

4.1. Factor and alternatives

The position of the fault in the code is considered as a factor, with alternatives being the number of faults in the first half of the code (FHC) and the number in the second half (SHC).

4.2. Hypotesis and variables

The statistical hypotheses for this second analysis use the visibility of the faults as the dependent variable, and for this a metric linked to the result of the individual fault detection process is used, the number of faults correctly detected in both sections of the code (FHC and SHC).

- H_{03}: The median of the errors detected in the FHC by the software engineering students is equal to the median of the errors detected in the SHC by said students.
- H_{13}: The median of the faults detected in the FHC by the software engineering students differs from the median of the faults detected in the SHC by said students.

4.3. Experimental design

The most appropriate experimental design for our study is the factorial design with a source of variation under a paired sample scheme (see Table 9), that is, two measurements are obtained from each of the 24 subjects, the number of failures correctly detected. in the FHC and the number of faults correctly detected in the FHC.

Table 9 –A source of variation with paired samples

Experimental subject	Measurement 1	Measurement 2
1	# Faults in the FHC	# Faults in the SHC
…	…	…
24	# Faults in the FHC	# Faults in the SHC

4.4. Descriptive Analisys

Table 10 presents some of the most important measures of central tendency and variability for the visibility of the faults variable. We can identify that the mean and median are higher with type roles 1, although the type of role with the least variability is 2; also, it is also possible to observe that the unbalanced sample contains fewer subjects with types 2 and 3.

Table 10 –Statistical summary for the visibility of the faults variable

Type	#	Mean	Median	SD
FHC	24	3.20833	3.0	1.28466
SHC	24	1.33333	1.0	1.46456

In Figure 5, we can observe that the FHC data sample has a bias to the left while the SHC data sample has greater symmetry. On the other hand, comparing both samples, quartile 1 of the FHC is above quartile 3 of the SHC, which leads us to think that there is a difference in the number of offenses detected in the FHC compared to the SHC. In this case, more faults are detected in the FHC.

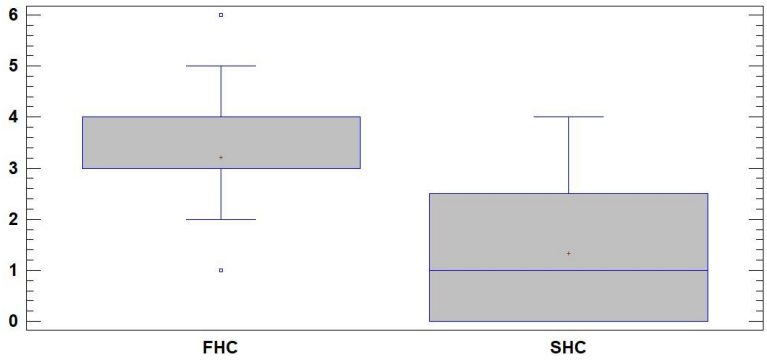

Figure 5 – Box and Whisker Plot for the visibility of the faults variable.

Figure 6 shows the results of the correct type faults by position, which were found by the subjects during the experiment.

Figure 6 – Faults by position.

In Figure 6, you can observed that most of the correct faults found are in the first half of the code, and one fault that was not found was the initialization fault.

4.5. Inferencial analisys

Since two paired samples were uncompared using an ordinal mean for the dependent variable, the Wilcoxon signed rank test was selected, which evaluates the null hypothesis that the median of PM-SM is equal to 0.0 in contrast to the alternative that the PM-SM median is not equal to 0.0

Table 11 –Wilcoxon signed rank test for the Variable Fault position

Metric	Ratio F	p-value
# of Faults	3.84157	0.000122292

Given that the p-value is less than 0.05, the null hypothesis is rejected and we can consider that there is a significant difference between the medians of the faults detected in the FHC with respect to the faults detected in the SHC (see Table 11).

5. EXPERIMENTAL ANALYSIS III: BY GENDER OF REVIEWER

The objective of this third analysis is to explore whether the gender of the reviewer has any influence on the detection of faults in the code. The correct number of faults detected in the code has been considered as a metric. As in the second analysis, it has been considered that because the failures are of different types, the metric is considered on an ordinal scale.

5.1. Factor and alternatives

The biological gender of the reviewer is considered as the source of variation, the alternatives being Male gender (MG) and female gender (FG).

5.2. Hypotesis and variables

The statistical hypotheses for this second analysis use the visibility of the faults as a dependent variable, and for this a metric linked to the result of the individual fault detection process is used, the number of faults correctly detected by the subjects based on their biological gender. (MG and FG),

- H_{04}: The median of the errors detected by the MG is equal to the median of the errors detected by the FG.
- H_{14}: The median of the faults detected by the MG differs from the median of the faults detected by the FG.

5.3. Experimental design

The most appropriate experimental design for our study is the factorial design with one source of variation and two alternatives (see Table 12), where the alternatives are: the male gender and the female gender.

Table 12 –Factorial design with one source of variation and two alternatives

Experimental Unit	1	2	3	…	24
Gender	MG	FG	FG	…	MG

5.4. Descriptive Analisys

Table 13 presents some of the most important measures of central tendency and variability for fault detection by gender. We can identify that the mean and median are higher for the female gender, and that the standard deviation is lower for this alternative.

Table 13 –Statistical summary for the faults detected by gender

Type	#	Mean	Median	SD
MG	14	4.28571	4.0	2.70124
FG	10	4.9000	4.5	1.91195

In the Figure 7 we can see that the female sample has less variability, but said range of values is included in the range of the male sample, so from a visual perspective, it seems that there are no significant differences between both alternatives.

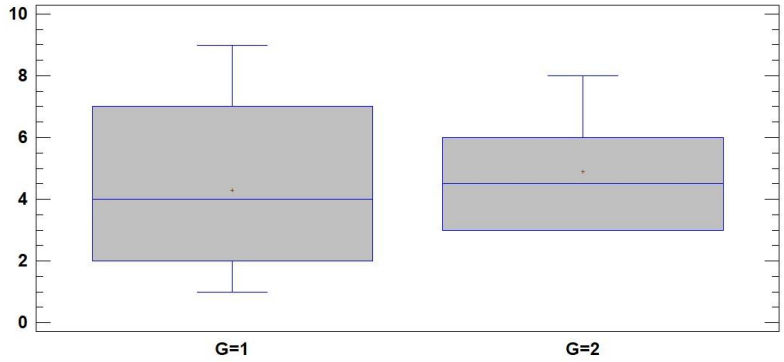

Figure 7 –Box and whisker plot for the faults detected variable.

5.5. Inferencial analisys

Since the purpose of the experimental design is to compare two alternatives from independent samples, the Mann Withney U test was selected.

Table 14 –W de Mann-Whitney test for faults detected variable

Metric	W	p-value
# of Faults	82.5	0.477348

Given that the p-value is greater than the value of the test's significance level (0.05), the null hypothesis is accepted, so it is possible to conclude that there is no significant difference between the faults detected by the male testers and those of the feminine gender (see Table 14).

6. CONCLUSIONS

In this paper, a first controlled experiment was presented with software engineering students with the objective of investigating whether one of the types of Belbin roles is compatible with the testing process using code review techniques, likewise, a deeper analysis investigating whether the position of the faults and the gender of the subjects show significant differences between the detected faults.

As a result, it was observed in the experiment that:

- do not show significant differences between the three types of role,
- show significant differences between the faults detected in the first half of the code with respect to the remaining second half,
- do not show a significant difference between the detected errors by gender.

It is worth mentioning that some subjects identified the correct fault in the code; however, they confused the type of fault. On the other hand, the majority of the correct faults found are found in the first positions of the code, this may be because the subjects did not have time to review the last lines of code.

Finally, some faults were not detected possibly because they were not designed as well as others (e.g. as mentioned in the previous paragraph, they caused confusion, forget, etc.)

The reflection on the part of the researchers forces them to carry out a short-term analysis of the faults injected into the code in order to obtain a better version of the experimental object, with which a second controlled experiment can be carried out with under less restrictive conditions and a larger sample.

REFERENCES

Aguilar, R., Peña, A., Díaz, J., & Ucán, J. (2022) Influence of Belbin's Role Theory on Database Design: Experimenting with Software Engineering Students. *Programming and Computer Software,* 48(8), 489-498.

Basili, V., & Perricone, B. (1984) Software errors and complexity: an empirical investigation. *Communicatons of the ACM*, 27(1), pp. 42-52.

Basili, V., & Selby, R. (1987). Comparing the effectiveness of software testing strategies, *IEEE transactions on software engineering*, (12), 1278-1296.

Belbin, M. (1981), *Management teams. Why they succeed or fail*. John Wiley & Sons.

Belbin, M. (1993). *Team roles at Work*. Elsevier Butterworth Heinemann.

Bourque, P., & Fairley, R. (2014). *Guide to the Software Engineering Body of Knowledge (SWEBOK V3.0)*. IEEE Computer Society.

Dreyfus, H. & Dreyfus, S. (1986) *Mind over Machine. The Power of Human Intuition and Expertise in the Era of the Computer*. Basil Blackwell.

Estrada, E., & Peña, A. (2013). Influencia de los roles de equipo en las actividades del desarrollador de software. *Revista Electrónica de Computación, Informática, Biomédica y Electrónica*, 2(1), 1-19.

Genero, M., Cruz-Lemus, J., & Piattini, M. (2014). *Métodos de investigación en ingeniería de software*. Ra-Ma.

Gutierrez, H., & De la Vara, R. (2012). *Análisis y Diseño de Experimentos*. McGraw Hill.

Juristo, N., & Moreno, A. (2001). *Basics of Software Engineering Experimentation*. Kluwer Academic Publishers.

Juristo, N. Moreno, A., & Vegas, S. (2006). *Técnicas de evaluación de software*. Universidad Politecnica de Madrid.

Morales, N., & Vega, V. (2018). Factores Humanos y la Mejora de Procesos de Software. Propuesta inicial de un catálogo que guíe su gestión. *Revista Ibérica de Sistemas y Tecnologías de Información*, (29), pp. 30-42.

Ucán, J., Aguilar, R., & Díaz, J. *Entornos Virtuales Inteligentes basados en Colaboración para Asistir el Aprendizaje de la Programación.* CONAIC.

Weigers, K. (1996). *Creating a Software Engineering Culture*. (pp. 28-34). Dorset House Publishing.

BIOGRAPHICAL NOTES

Juan Pablo Ucán Pech has a degree in Computer Science by Faculty of Mathematics of the Autonomous University of Yucatan, a Master in Computer Systems by Technological Institute of Merida, and Ph. D. in Computer Systems by the Postgraduate and Research Directorate of the University of the South, Mexico. He is currently a professor at the Faculty of Mathematics of the Autonomous University of Yucatan. He is a member of the Academic Body Research of Software Engineering for Education. He has the recognition of the National System of Researchers as a National Candidate Researcher by CONAHCYT and has the recognition of the Program for the Professional Development of Teachers (PRODEP) since 2014. His research work focuses on the areas of Software Engineering and Computer Science Educational.

Raúl Antonio Aguilar Vera obtained the degree of Ph. D. from the Polytechnic University of Madrid, Spain (European Doctor Mention) and the Master's degree in Software Engineering from the same Institution; he also has the degree of Master in Higher Education from the Autonomous University of Yucatan. He is currently a professor at the Faculty of Mathematics of the Autonomous University of Yucatan. He is responsible for the Academic Body Research of Software Engineering for Education. He has been a member since 2014 of the CONAHCYT National Researchers System (Level 1). His research work includes the areas of Software Engineering and Educational Computing.

Julio César Díaz Mendoza is an Industrial Engineer by Technological Institute of Merida, and has a Master's degree in Information Technology by Inter-American Development University. He is a member of the Academic Body Research of Software Engineering for Education. He is currently a professor at the Faculty of Mathematics of the Autonomous University of Yucatan and an Institutional Representative before the National Association of Computer Education Institutions (ANIEI). He has recognition from the Program for Teacher Professional Development (PRODEP). His research work includes the areas of Software Engineering and Educational Computing.

Antonio Armando Aguileta Güemez has a degree in Computer Science by Faculty of Mathematics of the Autonomous University of Yucatan, Ph. D. and Master in Computer Science by Institute of Technology and Higher Studies of Monterrey. He is currently a professor at the Faculty of Mathematics of the Autonomous University of Yucatan. He is a collaborator of the Academic Body Research of Software Engineering for Education. He has the recognition of the National System of Researchers as a National Candidate Researcher by CONAHCYT and has the recognition of the Program for the Professional Development of Teachers (PRODEP) since 2022. His research work includes the areas of Software Engineering and Educational Computing.