Evaluating a Language Workbench: from Working Memory Capacity to Comprehension to Acceptance

Giovanna Broccia*, Alessio Ferrari*, Maurice ter Beek*, Walter Cazzola†, Luca Favalli†, Francesco Bertolotti†
* ISTI–CNR, Pisa, Italy
{giovanna.broccia,alessio.ferrari,maurice.terbeek}@isti.cnr.it
†University of Milan, Italy
{walter.cazzola,luca.favalli,francesco.bertolotti}@unimi.it

***Abstract*—Language workbenches are tools that enable the definition, reuse and composition of programming languages and their ecosystem. This breed of frameworks aims to make the development of new languages easier and more affordable. Consequently, the comprehensibility of the language used in a language workbench (i.e., the meta-language) should be an important aspect to consider and evaluate. To the best of our knowledge, although the quantitative aspects of language workbenches are often discussed in the literature, the evaluation of comprehensibility is typically neglected.**

Neverlang is a language workbench that enables the definition of languages with a modular approach. This paper presents a preliminary study that intends to assess the comprehensibility of Neverlang programs, evaluated in terms of users' effectiveness and efficiency in a code comprehension task. The study also investigates the relationship between Neverlang comprehensibility and the users' working memory capacity. Furthermore, we intend to capture the relationship between Neverlang comprehensibility and users' acceptance, in terms of perceived ease of use, perceived usefulness, and intention to use. Our preliminary results on 10 subjects suggest that the users' working memory capacity may be related to the ability to comprehend Neverlang programs. On the other hand, effectiveness and efficiency do not appear to be associated with an increase in users' acceptance variables.

***Index Terms*—Program comprehension, Language workbench, Users study, Working memory capacity, Technology acceptance model**

I. Introduction

Language workbenches [1] are tools that support the development of general-purpose and domain-specific programming languages, their implementation, and their ecosystem including but not limited to a dedicated IDE. This breed of tools exploits the concepts of *language feature* and *component* [2] to modularise the specification and implementation of both the language and its ecosystem. Such a modularisation makes the definition and development of new languages easier and more affordable by reusing existing language components [3].

Most language workbenches are academic products that must be evaluated for comprehensibility, before being released to a wider public. However, while such instruments have been studied from a quantitative point of view [3]–[5], we are not aware of any literature that investigates the comprehensibility of language workbenches' meta-languages. This is important though, since language comprehension plays an essential role in the usage of the language itself, and it has an impact on user performance—-the time software developers spend on code comprehension is 30%–70% of their daily work time [6], [7].

This paper presents the first, preliminary study on the comprehensibility of a language workbench named Neverlang [8]. We select this tool as a representative case, as it shares most of the standard features of the other workbenches, it is a full-fledged framework with native support for language product lines [9], and it has been quite well analysed from a quantitative point of view [10]. We aim to analyse the comprehensibility of Neverlang from four main perspectives: 1) the effectiveness and efficiency of novice users in a code comprehension test; 2) the relation between effectiveness/efficiency and the *working memory capacity* of users, measured through standard tasks [11]—typically, users' performance is jeopardised when the mental effort required to comprehend the program exceeds their mental capacity [12]; 3) the extent to which users consider Neverlang acceptable, in terms of ease of use, usefulness and intention to use, following the Method Evaluation Model (MEM) [13]; 4) the relation between effectiveness/efficiency and degree of acceptance of the language. To this end, we conduct a pilot study with 10 subjects. The subjects first perform a test of their memory capacity, followed by a test of code comprehension, based on the template from Kosar et al. [14]. Finally, they are asked to fill out a questionnaire to evaluate acceptance-related variables.

Our preliminary results show that 1) Neverlang is overall comprehensible for users (average effectiveness is 79%); 2) the users' memory capacity could be a factor influencing the predisposition to language comprehension; 3) Neverlang is overall accepted by users (average scores of acceptance variables are all significantly higher than the median); 4) there is a negative relationship between the efficiency of users in the comprehension test and the acceptance variables.

This first comprehensibility study targeting language workbenches establishes an analysis framework that can be used to further evaluate Neverlang and other meta-languages. We make our replication package available [15].

II. Background and State of the Art

a) Neverlang: Neverlang [8] is a language workbench for the modular development of programming languages.

This work has been partially supported by the Italian MUR–PRIN 2020TL3X8X project T-LADIES: Typeful Language Adaptation for Dynamic, Interacting and Evolving Systems.

Language components, called slices, embody the concept of language features and are developed as separate units that can be independently compiled, tested, and distributed, enabling developers to share and reuse the same units across different language implementations. Each slice is composed of several modules. A module may contain a syntax definition and/or some roles. Each role defines a compilation phase by declaring semantic actions that should be executed when some syntax is recognised. Syntactic definitions and semantic roles are tied together using slices, which are composed to form a language.

Neverlang has been evaluated in the study of Cazzola et al. [10] by applying several metrics, like such as coupling and cohesion, to assess its ease of use in supporting variability in the production of families of domain-specific languages. To the best of our knowledge, no literature studies evaluate language workbenches in terms of meta-language comprehensibility, as most compare workbenches or define a taxonomy [3]–[5].

b) Working Memory Capacity: Working Memory is a cognitive system with limited capacity (referred to as *WM span* or *capacity*) responsible for the transient holding and processing of information necessary to complete a task [16]. WM capacity was proven to be highly predictive of performance in a number of different activities [17]–[21] and it is an important individual variable in general intellectual ability [22]–[24]. To measure such a capacity, one uses specific tasks, referred to as WM span tasks (WMST) [11]. These tasks entail performing two concurrent activities: one mnemonic which imposes the memorisation and recall of a set of elements (usually digits) and one processing which imposes the evaluation of elements (e.g., sentences or equations). Usually, to avoid bias and to assess the users' WM span more precisely, two different WMSTs are administered to users diversifying the processing activities (e.g., one task entailing mathematical processing and one entailing linguistic processing). The evaluation of the tasks is generally performed only for the mnemonic activity [25], [26], and a score is associated with the evaluation. In our study, we use the partial-credit unit (PCU) scoring method, favoured by empirical results [11]. PCU ranges from 0 to 1, an average PCU of 0.8 is considered high.

Regarding the evaluation of the cognitive aspects connected to program comprehension, Gonçales et al. [27], [28] propose a systematic mapping study of methods used to measure software developers' cognitive load (CL). They focus on CL measurement through sensors and study which kind of sensors are adopted, what metrics are used, and what algorithms are employed to classify CL. CL is the load imposed on the WM by a certain task [12]. For what concerns the methods, the eye tracker seems to be the more prominent one [29]–[33].

The measurement of CL through sensors during the execution of tasks, next to being invasive, also requires the presence of a moderator. On the contrary, we employ a non-invasive method to measure the WM capacity of users *before* performing the comprehension tasks, i.e., using WMSTs [11]. This allows us to classify users according to their cognitive performance, without the need to carry out the tests in presence of a moderator and thus allowing users to take the test when they prefer and when their cognitive abilities are at their best.

c) Technology Acceptance & Method Evaluation Models: The Technology Acceptance Model (TAM) [34] is a model to evaluate information technologies. According to the model, the users' usage of new technology is influenced by their perceived usefulness (PU) and their perceived ease of use (PEOU). These perception measures are combined with performance measures within the Method Evaluation Model (MEM) [13], for evaluating information systems design methods. The MEM incorporates the *actual* and the *perceived* success. The actual success is measured through two performance-based variables: effectiveness and efficiency; the perceived success, conversely, is measured through the combination of PU, PEOU, and a third perception-based variable, intention to use (ITU).

The TAM has been used in a wider set of applications: from the evaluation of e-learning systems [35], [36] to communication platforms [37], [38], to mention a few examples. The MEM was applied to the field of requirements engineering to predict the understandability of requirement models [39]. As far as we know, no previous research investigated the use of either of the two models for program comprehension. In this paper, we adapt both the TAM and the MEM to assess the comprehensibility of Neverlang, studying whether there exists a relation between users' acceptance and comprehensibility.

III. Experiment Design

Our goal is to answer the following four research questions:

RQ1. To what extent is Neverlang comprehensible in terms of syntax and semantics of its main constructs?

RQ2. Is Neverlang more comprehensible for users with a higher working memory capacity?

RQ3. To what extent is Neverlang accepted by users?

RQ4. Is there a relationship between Neverlang comprehensibility and its acceptance by users?

Comprehensibility (RQ1–RQ4) is evaluated through effectiveness (number of correct answers over the number of questions) and efficiency (effectiveness over time) [39], with a test questionnaire. The WM capacity is evaluated through two WMSTs: a reading span task, where users must process the veracity of sentences while memorising a set of numbers, and an operation span task, where users need to process the correctness of equations while memorising a set of numbers. User acceptance (RQ3–RQ4) is evaluated through three questionnaires aiming to assess three perception-based variables, viz. PEOU, PU, and ITU.

The experiment is composed of the following four phases:

- **WMST phase:** Users are asked to perform the two WMSTs on an online platform. The order in which the two tasks are presented is randomly set by the platform.
- **Training phase:** Users are asked to watch a video about Neverlang. The video lasts less than 15 minutes and gives users all the necessary information to perform the test phase. Users are asked to watch the video before the test, preferably once. However, they can use this support during the test in case of need, provided they report the number of times they did so.

- **Test phase:** Users are asked to answer eight questions about Neverlang on an online survey platform. The questions are divided according to two comprehensibility aspects: three questions for *learnability* and five questions for *understandability*. This phase is organised according to the study illustrated in the paper by Kosar et al. [14] The test was defined in collaboration with two Neverlang experts, who considered it sufficiently effective to evaluate the comprehension of the language by novices. Each question in the test phase is a multiple-choice question and can contain a (set of) Neverlang statement(s), a description of a (set of) statement(s), or a simple request (e.g., select the set of Neverlang statements that satisfy a given characteristic). Each question contains five choices in terms of sets of statements, statements' meanings, or given answers. Effectiveness is computed as the average of correct answers divided by the number of questions. Efficiency is computed as the effectiveness divided by the time spent answering all questions.
- **Post-study phase:** Users are asked to fill out an online questionnaire containing a number of questions about their prior knowledge (on programming and language workbenches), personal details (age, gender, occupation), and feedback on the test and on the training material. The questionnaire also includes three different 5-point scale questionnaires on PEOU, PU, and ITU, consisting of 8, 14, and 8 statements, for each variable, respectively. The questionnaires consist of a shuffled set of positive statements about Neverlang (e.g., *Neverlang is easy to learn*) and their negation (*Neverlang is NOT easy to learn*), to prevent systematic response bias. Each statement requires an answer on a 5-point scale, from 1 (strongly disagree) to 5 (strongly agree). Points for negation statements are counted as 6 minus the point given as the answer. We compute the value for each variable as the average of the answer points for the statements associated with the variable. This design follows that from Abraho et al. [39].

In order not to overload their cognitive effort, we asked participants to perform the WMST phase and the remaining phases on two separate days. All the material, including the platform for WMSTs and the video, is available [15].

IV. Execution and Results

a) Subject Selection: We recruited 10 subjects from the University of Milan. The subjects are bachelor students (5), master students (4), and graduate students (1) in Computer Science. Their skills in programming and in object-oriented programming, is self-evaluated with an average score of 3.7 on a scale from 1 to 5. More than 70% of the subjects knew neither Neverlang nor any other language workbenches before the test. Therefore, our preliminary results apply to subjects that have a similar background, and different outcomes may be observed with participants with different characteristics.

b) Analysis Procedure: To answer RQ1, we evaluate the results of the tests in terms of effectiveness and efficiency and we compare the values with target ones established by the Neverlang experts involved in the study. Effectiveness greater than 0.6 (60% of the maximum value equal to 100% of the questions correctly answered) is considered to indicate sufficient comprehension. Since the Neverlang test is estimated to last 18 minutes, and considering that the maximum effectiveness is 1, the target efficiency is 0.06. An efficiency of 0.36, i.e., 60% of the target, can be considered adequate. We also check whether there is a relationship between effectiveness and efficiency by fitting a linear model. To answer RQ2, we fit a linear model between PCU and effectiveness, and between PCU and efficiency. For RQ3, we perform a Wilcoxon signed-rank test to check if PEOU, PU, and ITU are significantly higher than the median score (i.e., 3 = neither agree nor disagree). Finally, for RQ4, we fit linear models of PEOU, PU, and ITU with respect to effectiveness and efficiency.

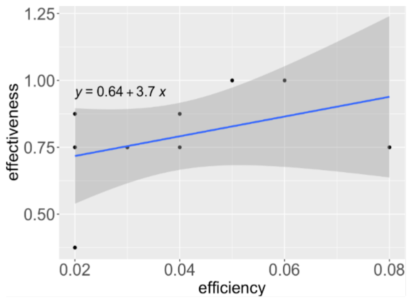
c) Results: Table II shows the descriptive statistics of the different variables. Table I reports the linear models fitted on each relevant variable pair, and Figure 1 shows the linear models, together with the boxplots of PEOU, PU, and ITU. Many of the relationships captured by the linear models are not statistically significant. However, some general tendencies, reported below, can be observed even at this preliminary stage.

RQ1. The effectiveness is ~ 0.79 on average, meaning that 79% of the questions of the test are correctly answered. Based on the 0.6 target, this suggests a more than sufficient degree of comprehensibility, especially considering that the participants only had a 15-minute training. The average efficiency is 0.04, so 67% of the target efficiency value 0.06 (cf. above). Also according to this dimension, Neverlang can be considered sufficiently comprehensible, although there is room for improvement. Looking at the relationship between effectiveness and efficiency (Fig. 1a), we see a positive linear relation, although not significant (p-value ~ 0.24, cf. Table I), possibly due to the limited number of subjects.

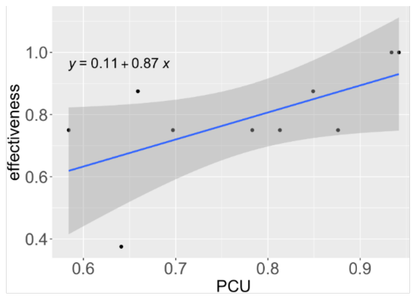
RQ2. Concerning the relationship between the users' WM capacity and effectiveness/efficiency, Figs. 1b and 1c show a positive correlation between these measures, confirming that the WM capacity is correlated with the predisposition

TABLE I
Statistics from Fitting Linear Models Considering Relevant Variable Pairs (x and y Columns).
* indicates that results are weakly significant (p-value < 0.1)
** indicates that results are significant (p-value < 0.05)

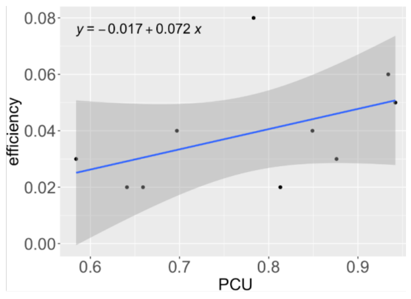
y	x	**Eq.**	R^2	p-**value**
efficiency	effectiveness	$y = 0.64 + 3.7x$	0.1679	0.2395
PCU	effectiveness	$y = 0.11 + 0.87x$	0.3858	0.0553*
PCU	efficiency	$y = -0.017 + 0.072x$	0.2123	0.1802
PEOU	effectiveness	$y = 3.4 + 0.32x$	0.006676	0.8225
PU	effectiveness	$y = 4 + 0.02x$	8.91E-05	0.9794
ITU	effectiveness	$y = 3.1 + 0.54x$	0.03228	0.6194
PEOU	efficiency	$y = 4.6 - 23x$	0.4295	0.039**
PU	efficiency	$y = 4.5 - 12x$	0.391	0.0532*
ITU	efficiency	$y = 4.2 - 17x$	0.3785	0.0583*



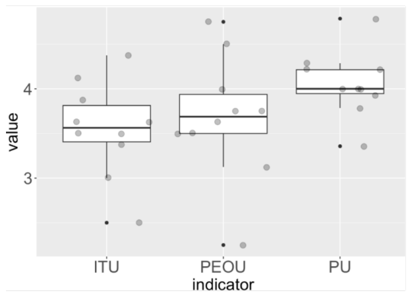
a) Relation between effectiveness and efficiency



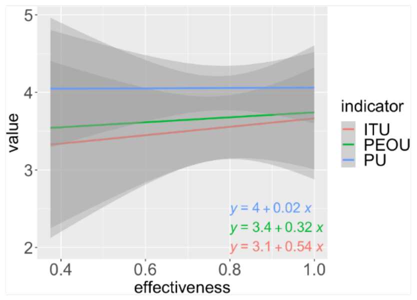
b) Relation between effectiveness and PCU



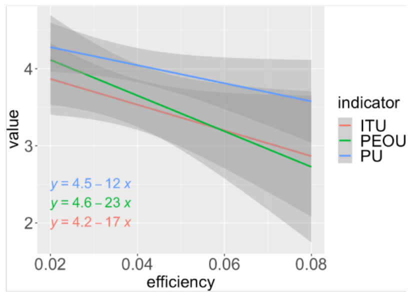
c) Relation between efficiency and PCU



d) Boxplot of users acceptance variables



e) Relation between users acceptance and effectiveness



f) Relation between users acceptance and efficiency

Fig. 1. Results charts

to language comprehension, since a high WM capacity has an impact on both effectiveness and efficiency. Concerning statistical significance, Table I shows that the relation between the PCU and effectiveness is weakly significant (p-value ~ 0.06), while the relation with efficiency is not significant.

RQ3. The results of the Wilcoxon signed-rank test indicate that the average scores for PEOU, PU, and ITU are all significantly higher than the median for $\alpha = 0.05$, with p-values ~ 0.02, ~ 0.003, and ~ 0.012, respectively. Therefore, we argue that Neverlang is positively considered by users in terms of acceptance. From the boxplot in Fig. 1d, we see that PU has high average values (median 4), while PEOU and ITU are slightly lower. This suggests that the Neverlang language requires further adjustments in terms of ease of use. In future work, we plan to carry out interviews with users to better understand which aspects they find most difficult to use and to understand. To increase ITU, we argue that we need to work more on making users understand the usage scenarios of Neverlang and, more in general, of the language workbenches. This was hardly feasible by means of a 15-minute video, and a longer overview appears to be required.

RQ4. Figure 1e shows the relation between the user acceptance variables and the effectiveness. The chart shows that there is a slightly positive relationship between the variables: essentially, the users that performed best in the test, tend to evaluate the language more positively in terms of acceptance. This relationship is however not significant, as shown in Table I. On the contrary, more efficient users show lower acceptance, as Fig. 1f shows. This negative relationship also shows a higher degree of significance (cf. last rows in Table I).

V. Threats to Validity

Concerning *construct validity*, to measure comprehensibility we used widely used metrics of effectiveness and efficiency [13], [39]. The test questionnaire was adapted from the template of Kosar et al. [14] and revised by two Neverlang experts, the WMSTs from Conway et al. [11], and the users' acceptance questionnaires used in the post-study phase were adapted from the work of Abrahao et al. [39]. We made an effort to maximise *internal validity*, by evaluating different quantitative and objective variables. Subjective opinions were collected with the three questionnaires, which could have led to participant response bias. However, the participants did not meet the experimenter, as the whole study was conducted online, thereby limiting the possible tendency of participants to provide positive answers to please the researchers. This paper focuses on the relationship between WMC and understandability. Other confounding factors, like user expertise, which might affect understandability, are not considered. However, we highlight that this work measures correlation, not causation.

TABLE II
Descriptive Statistics

Measures	*Median*	*Mean*	*Std. dev.*	*Min.*	*Max.*
PCU	0.798	0.778	0.116	0.584	0.942
Effectiveness	0.75	0.788	0.215	0.375	1
Efficiency	0.03	0.04	0.015	0.02	0.08
PEOU	3.688	3.675	0.697	2.25	4.75
PU	4	4.057	0.318	3.357	4.786
ITU	3.563	3.55	0.51	2.5	4.375

We also underline that this is not an experiment meant to compare two language workbenches. Therefore, we do not investigate the understandability of Neverlang in comparison with other platforms. Given the limited number of participants, and the contrived nature of the experiments, *external validity* is limited [40], and different results may be observed in daily practice with Neverlang. Furthermore, here we evaluate Neverlang in a code reading task. Different results might be observed if a coding task were considered.

VI. Conclusion and Future work

We presented the first comprehensibility study of Neverlang, a language workbench for modular development of programming languages, by conducting a pilot test with subjects from academia (10 computer science students of different levels). The preliminary results are promising. In the future, next to carrying out user interviews to grasp their difficulties, we plan to perform the test on a larger number of subjects in order to confirm or reject our preliminary results, check how particular cognitive dimensions contribute to the questionnaires' success, and compare Neverlang with other workbenches.

References

[1] M. Fowler. (2005, May) Language workbenches: The killer-app for domain specific languages? Martin Fowler's Blog. [Online]. Available: http://www.martinfowler.com/articles/languageWorkbench.html

[2] W. Cazzola *et al.*, "μ-DSU: A micro-language based approach to dynamic software updating," *Comput. Lang. Syst.*, vol. 51, pp. 71–89, 2018.

[3] S. Erdweg *et al.*, "The state of the art in language workbenches: Conclusions from the language workbench challenge," in *SLE 2013*, ser. LNCS, vol. 8225. Springer, 2013, pp. 197–217.

[4] ——, "Evaluating and comparing language workbenches: Existing results and benchmarks for the future," *Comput. Lang. Syst.*, vol. 44, pp. 24–47, 2015.

[5] S. Kelly, "Empirical comparison of language workbenches," in *Workshop on Domain-Specific Modeling (DSM)*. ACM, 2013, pp. 33–38.

[6] R. Minelli, A. Mocci, and M. Lanza, "I know what you did last summer: An investigation of how developers spend their time," in *Int. Conf. on Program Comprehension (ICPC)*. IEEE, 2015, pp. 25–35.

[7] X. Xia *et al.*, "Measuring program comprehension: A large-scale field study with professionals," *IEEE Trans. Softw. Eng.*, vol. 44, no. 10, pp. 951–976, 2017.

[8] E. Vacchi and W. Cazzola, "Neverlang: A framework for feature-oriented language development," *Comput. Lang. Syst.*, vol. 43, no. 3, pp. 1–40, 2015.

[9] D. Méndez-Acuña *et al.*, "Leveraging Software Product Lines Engineering in the development of external DSLs: A systematic literature review," *Comput. Lang. Syst.*, vol. 46, pp. 206–235, 2016.

[10] W. Cazzola and L. Favalli, "Towards a recipe for language decomposition: quality assessment of language product lines," *Empir. Softw. Eng.*, vol. 27, no. 4, pp. 82:1–82:47, 2022.

[11] A. R. Conway *et al.*, "Working memory span tasks: A methodological review and user's guide," *Psychon. Bull. Rev.*, vol. 12, no. 5, pp. 769–786, 2005.

[12] J. Sweller, "Cognitive load theory," in *Cognition in Education*, ser. Psychol. Learn. Motiv., J. P. Mestre and B. H. Ross, Eds. Academic Press, 2011, vol. 55, ch. 2, pp. 37–76.

[13] D. L. Moody, "Dealing with complexity: a practical method for representing large entity relationship models," Ph.D. dissertation, University of Melbourne, 2001.

[14] T. Kosar, M. Mernik, and J. C. Carver, "Program comprehension of domain-specific and general-purpose languages: comparison using a family of experiments," *Empir. Softw. Eng.*, vol. 17, pp. 276–304, 2012.

[15] G. Broccia *et al.* (2023, February) Evaluating a Language Workbench: from Working Memory Capacity to Comprehension to Acceptance – Supplementary Material. [Online]. Available: https://doi.org/10.5281/zenodo.7624649

[16] G. A. Miller, "The magical number seven, plus or minus two: Some limits on our capacity for processing information," *Psychol. Rev.*, vol. 63, no. 2, p. 81, 1956.

[17] M. Daneman and P. A. Carpenter, "Individual differences in working memory and reading," *J. Verb. Learn. Verb. Behav.*, vol. 19, no. 4, pp. 450–466, 1980.

[18] A. Miyake, M. A. Just, and P. A. Carpenter, "Working memory constraints on the resolution of lexical ambiguity: Maintaining multiple interpretations in neutral contexts," *J. Mem. Lang.*, vol. 33, no. 2, pp. 175–202, 1994.

[19] P. Barrouillet, "Transitive inferences from set-inclusion relations and working memory," *J. Exp. Psychol. Learn. Mem. Cogn.*, vol. 22, no. 6, p. 1408, 1996.

[20] P. C. Kyllonen and R. E. Christal, "Reasoning ability is (little more than) working-memory capacity?!" *Intell.*, vol. 14, no. 4, pp. 389–433, 1990.

[21] V. J. Shute, "Who is likely to acquire programming skills?" *J. Educ. Comput.*, vol. 7, no. 1, pp. 1–24, 1991.

[22] A. R. Conway *et al.*, "A latent variable analysis of working memory capacity, short-term memory capacity, processing speed, and general fluid intelligence," *Intell.*, vol. 30, no. 2, pp. 163–183, 2002.

[23] A. R. Conway, M. J. Kane, and R. W. Engle, "Working memory capacity and its relation to general intelligence," *Trends Cogn. Sci.*, vol. 7, no. 12, pp. 547–552, 2003.

[24] R. W. Engle *et al.*, "Working memory, short-term memory, and general fluid intelligence: a latent-variable approach," *J. Exp. Psychol. Gen.*, vol. 128, no. 3, p. 309, 1999.

[25] M. J. Kane *et al.*, "The generality of working memory capacity: a latent-variable approach to verbal and visuospatial memory span and reasoning," *J. Exp. Psychol. Gen.*, vol. 133, no. 2, p. 189, 2004.

[26] G. S. Waters and D. Caplan, "The measurement of verbal working memory capacity and its relation to reading comprehension," *Q. J. Exp. Psychol. A*, vol. 49, no. 1, pp. 51–79, 1996.

[27] L. Gonçales *et al.*, "Measuring the cognitive load of software developers: A systematic mapping study," in *Int. Conf. on Program Comprehension (ICPC)*. IEEE, 2019, pp. 42–52.

[28] L. J. Gonçales, K. Farias, and B. C. da Silva, "Measuring the cognitive load of software developers: An extended systematic mapping study," *Inf. Softw. Technol.*, vol. 136, 2021.

[29] A. Abbad-Andaloussi, T. Sorg, and B. Weber, "Estimating developers' cognitive load at a fine-grained level using eye-tracking measures," in *Int. Conf. on Program Comprehension (ICPC)*. ACM, 2022, pp. 111–121.

[30] J. P. Menzen, K. Farias, and V. Bischoff, "Using biometric data in software engineering: a systematic mapping study," *Behav. Inf. Technol.*, vol. 40, no. 9, pp. 880–902, 2021.

[31] U. Obaidellah, M. Al Haek, and P. C.-H. Cheng, "A survey on the usage of eye-tracking in computer programming," *ACM Comput. Surv.*, vol. 51, no. 1, pp. 1–58, 2018.

[32] Z. Sharafi *et al.*, "Eye-tracking metrics in software engineering," in *Asia-Pacific Software Engineering Conf. (APSEC)*. IEEE, 2015, pp. 96–103.

[33] Z. Sharafi, Z. Soh, and Y.-G. Guéhéneuc, "A systematic literature review on the usage of eye-tracking in software engineering," *Inf. Softw. Technol.*, vol. 67, pp. 79–107, 2015.

[34] F. D. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology," *MIS Q.*, vol. 13, no. 3, pp. 319–340, 1989.

[35] D. Al-Fraihat *et al.*, "Evaluating e-learning systems success: An empirical study," *Comput. Hum. Behav.*, vol. 102, pp. 67–86, 2020.

[36] S. Sukendro *et al.*, "Using an extended Technology Acceptance Model to understand students' use of e-learning during Covid-19: Indonesian sport science education context," *Heliyon*, vol. 6, no. 11, 2020.

[37] D. Pal and V. Vanijja, "Perceived usability evaluation of Microsoft Teams as an online learning platform during COVID-19 using system usability scale and technology acceptance model in India," *Child Youth Serv. Rev.*, vol. 119, 2020.

[38] H. A. Alfadda and H. S. Mahdi, "Measuring students' use of Zoom application in language course based on the technology acceptance model (TAM)," *J. Psycholinguist. Res.*, vol. 50, no. 4, pp. 883–900, 2021.

[39] S. Abrahão *et al.*, "Evaluating requirements modeling methods based on user perceptions: A family of experiments," *Inf. Sci.*, vol. 181, no. 16, pp. 3356–3378, 2011.

[40] K.-J. Stol and B. Fitzgerald, "The ABC of software engineering research," *ACM Trans. Softw. Eng. Methodol.*, vol. 27, no. 3, pp. 1–51, 2018.